DIPOG-2.1

User Guide
Direct Problems for Optical Gratings over Triangular Grids

A. Rathsfeld

Weierstraß-Institut für
Angewandte Analysis und Stochastik
Leibniz-Institut im Forschungsverbund Berlin e.V.

Mohrenstr. 39
D-10117 Berlin
Germany

rathsfeld@wias-berlin.de

Weierstrass Institute for
Applied Analysis and Stochastics

Abstract[1]

This guide describes how to use the programs FEM_CHECK, GFEM_CHECK, FEM, GFEM, FEM_PLOT, GFEM_PLOT, FEM_FULLINFO, GFEM_FULLINFO, and OPTIMIZE of the package DIPOG-2.1. The package is a collection of finite element (FEM) programs to determine the efficiencies of the diffraction of light by a periodic grating structure. It is based on the software package PDELIB and solves the classical case of TE and TM polarization and the case of conical diffraction. The code provides a conventional FEM and a generalized FEM (called GFEM). The latter is the variational approach of the conventional FEM combined with a new trial space. Other routines of DIPOG-2.1 determine optimal gratings of certain grating classes minimizing objective functionals depending on the efficiencies.

We note that the DIPOG-2.1 programs require the installation of the previous version DIPOG-1.3 or DIPOG-1.5, of the grid generator TRIANGLE-1.4, of the graphical user interface package FLTK, and of the equations solver PARDISO[2]. Additionally, some of them need the graphical package openGL (or the MESA emulation of openGL) together with GLTOOLS-2.4 or, alternatively, the package GNUPLOT. Examples of data and output files are enclosed.

[1] Don't read the complete user guide. Don't you have anything better to do? If you have to compute the efficiencies, phase sifts, and energies of the waves diffracted by gratings, then go to the directory DIPOG-2.1/GUI and start the program DIPOG-2.1-GUI which is self explanatory. Alternatively, for gratings with more complex input data, you should change to the directory DIPOG-2.1/CLASSICAL or DIPOG-2.1/CONICAL. Read the data file "example.dat". Change it according to your requirements. Run one of the executables with "example.dat" as argument. Read the results. Plots with efficiency curves can be produced via the executables in the directory DIPOG-2.1/RESULTS. If you still have a question, come back to this user guide and read the corresponding part, only. Good luck!

In case you have to optimize a grating, read the Sections 10.1.1, 10.1.2, 10.2, 10.3.1, and 10.3.2 of this user guide. Then go to the directory DIPOG-2.1/OPTIM and read the data file "example.dat". Change it according to your requirements. Run the executable OPTIMIZE with "example.dat" as argument. Read the results. If you still have a question, come back to this user guide and read the corresponding part, only. Good luck!

[2] PARDISO itself requires some routines from LAPACK and some BLAS routines.

Contents

1 Introductory Remarks and the Structure of the Package

1.1 What is DIPOG-2.1 and Dipog-1.5?

`DIPOG`-2.1 is a finite element (FEM) program to determine the efficiencies of the diffraction of light by a periodic grating structure. The unbounded domain is treated by coupling with boundary element methods. `DIPOG`-2.1 solves the classical TE and TM cases, i.e. the cases of incident light in the plane perpendicular to the grooves of the periodic grating, and the case of conical diffraction, i.e. of oblique incidence of light. The code is based on the package `PDELIB` which is a collection of software components to create simulators based on partial differential equations:

http://www.wias-berlin.de/software/pdelib

`DIPOG`-2.1 provides a conventional FEM approach as well as a generalized FEM version called GFEM. The latter is nothing else than the variational approach of the conventional FEM combined with a new trial space for the approximation of the unknown solution. To compute follow the subsequent instructions.

The earlier version `Dipog`-1.5 does the same for the special case of binary (lamellar) gratings[3], i.e. if the different grating material pieces are of rectangular shape with sides parallel to the axes. Whenever the user is confronted with binary grating geometries, he can use `Dipog`-1.5 or `DIPOG`-2.1. However, he should prefer the more efficient `Dipog`-1.5. The fast generalized FEM used in `Dipog`-1.5 cannot be applied to general polygonal geometries.

Beside the simulation of the diffraction by gratings, `DIPOG`-2.1 can optimize a few classes of grating structures. Local gradient based optimization methods and the global simulated annealing method is used, to determine optimal geometry parameters and optimal refractive indices, respectively. The algorithms work for classical and conical diffraction. The geometry classes admissible for our optimization are:

- gratings with a polygonal profile separating the cover and substrate material
- special classes of multilayered trapezoidal gratings
- fixed grating structures, where a part of a polygonal interface is to be optimized

Note that, for the classical case, `Dipog`-1.5 computes optimal binary gratings for given efficiency sequences or for prescribed energy restrictions. For `Dipog`-1.5, we refer to the German Benutzer-Handbuch:

http://www.wias-berlin.de/software/DIPOG

1.2 Programming language and used packages

All programs are written in fortran, c or c++ language and based on the `UNIX` system. The programs require the previous version `DIPOG-1.3` or `DIPOG-1.5`, the grid generator `TRIANGLE-`1.4, the graphical user package `FLTK`, and the linear equations solver `PARDISO` together with some `LAPACK` and `BLAS` routines. For good visualization the package `openGL` (or at least

[3]Contrary to the original meaning of binary, several layers are admitted.

the MESA emulation of openGL) is needed together with the auxiliary package GLTOOLS. A minor visualization is possible with the program package GNUPLOT. In emergency case, the computations run also without any visualization, i.e. without openGL and GNUPLOT.

For non-comercial use, the Levenberg-Marquardt algorithm levmar-2.2 by Manolis Lourakis can be applied for optimization. The package DIPOG-2.1 can use the refractive indices from the tables of the program package IMD (cf. Section 4). To this end all the files from the IMD directory "imd/imd/nk.dir" must be copied to the DIPOG-2.1 directory "refr_ind_data". Of course, the user must observe the copyrights of the package IMD.

To get informations on the above mentioned necessary packages, we refer to:

DIPOG:	http://www.wias-berlin.de/software/DIPOG
TRIANGLE:	http://www.cs.cmu.edu/∼ quake/triangle.html
GLTOOLS:	http://www.wias-berlin.de/software/gltools
GNUPLOT:	http://www.gnuplot.info
FLTK:	http://www.fltk.org
IMD:	http://cletus.phys.columbia.edu/∼windt/idl
levmar:	http://www.imm.dtu.dk/pubdb/views/edoc_download.php/3215/pdf/imm3215.pdf
PARDISO:	http://www.computational.unibas.ch/cs/scicomp/software/pardiso

1.3 Get executables, comment lines in input files

If the executable programs do not exist, then generate them by using the file "makefile" located in the DIPOG-2.1 home directory. To do so, produce a header file MHEAD in the subdirectory MAKES of the DIPOG-2.1 home directory. Just copy one of the example files MHEAD_SGI, MHEAD_LINUX, or MHEAD_DEC to MHEAD and change the operating system, the paths, and the flags according to your computer system.[4] Then go to the home directory

```
cd DIPOG-2.1
```

and add the commands:

```
make clean
make
```

If the package is installed, then the programs can be used by different users simultaneously. To this end, each user should have a private DIPOG-2.1 home directory containing the first six subdirectories. These subdirectories together with all data and example files and with the correct links to the executables can be created automatically in the chosen new private home directory by calling the executable MAKEHOME. Note that MAKEHOME has just been created by "make" in the subdirectory MAKES of the installed package. Before a user runs the executables, he has to set the environment variable LD_LIBRARY_PATH such that the directory containing PARDISO is included. Setting the environment variable OMP_NUM_THREADS to a non-negative integer, he limits the number of used CPUs. The solver routine PARDISO runs parallel.

[4]The temporary directory is defined in MHEAD before the installation. Its name will be stored in the first line of the file MAKES/make_info and can be changed in this file at any time. Alternatively, the temporary working directory can be chosen by setting the environment variable TMPDIR.

Most of the subsequent executables can be called without argument. Then one gets information on the necessary arguments. Usually, all input files contain a lot of explanations and informations. Indeed, each line beginning with the sign "#" is a comment. Such lines can be added or deleted without any problem.

1.4 Structure of the package

The directory containing the `README`.txt file is the home directory of `DIPOG`-2.1. We suppose in this user guide that it is named `DIPOG`-2.1. There exist the following important subdirectories:

`GEOMETRIES` → input files "name.inp"
(geometrical data of the gratings),
executable `SHOW`
(to visualize the input data "name.inp", exists only with `openGL`, argument: "name.inp")
executable `TGUI`
(graphical user interface to create input files "name.inp")
executable `GEN_INPUT`
(to generate a general input file, no argument)
executable `GEN_ECHELLEA`
(to generate an input file for an echelle grating of type A,
first argument: name without tag "inp" of file to be created,
second argument: letter A,L,R,
third argument: depth/angle,
fourth argument: width of first part of layer,
fifth argument: width of second part of layer)
executable `GEN_ECHELLEB`
(to generate an input file for an echelle grating of type B,
first argument: name without tag "inp" of file to be created,
second argument: angle, third argument: width of layer)
executable `GEN_ECHELLE`
(to generate an input file for a general echelle grating,
first argument: name without tag "inp" of file to be created,
second argument: letter "A" for apex angle, "R" for right blaze angle, "L" for left blaze angle or "D" for depth,
third argument: angle/depth,
fourth argument: letter "A" for apex angle, "R" for right blaze angle, "L" for left blaze angle or "D" for depth,
fifth argument: angle/depth,
sixth argument: width of layer over left blaze side,
seventh argument: width of layer over right blaze side)
executable `GEN_TRAPEZOID`
(to generate an input file for a trapezoidal grating, first argument: name without tag "inp" of file to be created,
second argument: angle, third argument: length of basis,

fourth argument: number of material layers in trapezoid,
next arguments: heights of material layers, last argument:
height of coating layer)
executable `GEN_MTRAPEZOID`
(to generate an input file for a grating with several trapezoids,
one beside the other, first argument: name without tag
"inp" of file to be created, second argument: name of input
file "mtrapezoid.INP" containing data of grating)
executable `GEN_LAMELLAR`
(to generate an input file for a lamellar grating, first
argument: name without tag "inp" of file to be created,
second argument: name of input file "lamellar.INP"
containing location and widths of layers)
file lamellar.INP
(to define location and widths of layers in grating
generated by `GEN_LAMELLAR`)
executable `GEN_POLYGON`
(to generate an input file for a grating with polygonal profile
curve, first argument: name without tag "inp" of file to be
created, second argument: name "file1" of file with
nodes of polygon)
executable `GEN_POLYGON2`
(to generate an input file for a grating with polygonal
profile curve and with coating, first argument: name without
tag "inp" of file to be created, second argument: name
"file1" of file with nodes of polygon, third argument:
name "file2" of file with nodes of boundary
of coated layer)
file file1
(to define profile line for a polygonal grating generated by
`GEN_POLYGON` or `GEN_POLYGON2`)
file file2
(to define polygonal boundary line for the coated layer
of polygonal grating generated by `GEN_POLYGON2`)
executable `GEN_PROFILE`
(to generate an input file for a profile grating given
by c-code, first argument: name without tag "inp" of file
to be created, second argument: stepsize of polygonal
approximation)
c-code file profile.c
(to define profile line for a profile grating of `GEN_PROFILE`)
executable `GEN_PROFILES`
(to generate an input file for a grating given by many
profile lines defined by c-code, first argument: name without
tag "inp" of file to be created, second argument: stepsize of

polygonal approximation)
c-code file profiles.c
(to define profile lines for a profile grating of `GEN_PROFILES`)
executable `GEN_PIN`
(to generate an input file for a pin grating given by a profile line defined by c-code, first argument:
name of input file to be created, second argument: stepsize of polygonal approximation)
c-code file pin.c
(to define the profile line for a pin grating of `GEN_PIN`)
executable `GEN_CPIN`
(to generate an input file for a coated pin grating given by profile lines defined by c-code, first argument:
name of input file to be created, second argument:
stepsize of polygonal approximation)
c-code file cpin.c
(to define profile lines for a coated pin grating of `GEN_CPIN`)
executable `POINT_CLOUD_INP`
(to generate an input file for an advanced grating with point clouds to enforce a mesh grading at the corner points, first argument: name without tag "inp" of file to be improved, second argument: number of cloud points at circular layer, third argument: number of layers, fourth argument: threshold angle for corners with point clouds, output: geometry input file with a "+" added to the name before the tag ".inp")
executable `GEN_CPIN2`
(to generate an input file for a coated
pin grating of type 2 given by
profile lines defined by c-code, first argument:
name of input file to be created, second argument:
stepsize of polygonal approximation)
c-code file cpin2.c
(to define profile lines for a coated pin grating of `GEN_CPIN2`)

`CLASSICAL` → input files "name.dat"
(non-geometrical data of the gratings),
data file "generalized.Dat"
(data for the `GFEM`),
executables `FEM` and `GFEM`
(for simple calculation, case of classical diffraction,
argument "name.dat"),
executables `FEM_CHECK`, `GFEM_CHECK`
(for check of input, exists only with `openGL`,
argument "name.dat"),
executables `FEM_PLOT` and `GFEM_PLOT`

(for calculation with plots of resulting fields, case of classical diffraction, exists only with `openGL` or `GNUPLOT`, argument “name.dat”),
executable `GFEM_MATLAB`
(for calculation with plots of resulting fields, case of classical diffraction, plots in form of `Matlab` file, argument “name.dat”),
executables `FEM_FULLINFO` and `GFEM_FULLINFO`
(for calculation with additional information, case of classical diffraction, argument “name.dat”)
executable `GFEM_MOVIE`
(creates movie of z-coordinate of fields depending on time, case of classical diffraction, movie in form of `Matlab` file, argument “name.dat”),

`CONICAL` → input files “name.dat”
(non-geometrical data of the gratings),
data file “conical.Dat”
(data for the `GFEM`),
executables `FEM` and `GFEM`
(for simple calculation, case of conical diffraction, argument “name.dat”),
executables `FEM_CHECK`, `GFEM_CHECK`
(for check of input, case of conical diffraction, exists only with `openGL`, argument “name.dat”),
executables `FEM_PLOT` and `GFEM_PLOT`
(for calculation with plots of resulting fields, case of conical diffraction, exists only with `openGL` or `GNUPLOT`, argument “name.dat”),
executables `FEM_FULLINFO` and `GFEM_FULLINFO`
(for calculation with additional information, case of conical diffraction, argument “name.dat”)

`OPTIM` → input files “name.dat”
(data of the gratings and optimization),
data file “conical.Dat”
(data for the generalized finite elements),
executable `OPTIMIZE`
(using various flags, this does all the work: check data, check gradients, plot gradients, optimize grating, plot solution of optimization)
executable `CONVTEST`
(runs a convergence tests with initial solution set to corners of box domain defining the constraints, input file is the same as for `OPTIMIZE`, but the initial solution of input file must be the exact solution)
executable `OPTIM2OPTIM`

(extracts new data input file for OPTIMIZE from old input file with reduced number of data in objective functional)
executable SHOWMEAS
(creates plots of data used in objective functional, input is data input file for OPTIMIZE)
executable CLASSIC2OPTIM
(extracts test data input file for OPTIMIZE from input and result file of GFEM in CLASSICAL)
executable SENSITIVITY
(extracts optimal sets of data for quadratic objective functional in OPTIMIZE, input files are the data file for OPTIMIZE and files with Jacobians generated by OPTIMIZE)
executable OPTIM2JACOBIAN
(alternative executable to produce files with Jacobians for SENSITIVITY)
executable NOISETEST
(executable to test the dependency of the optimization result on noisy data)

GUI → input files "name.dat"
(non-geometrical data of the gratings, like in CLASSICAL and CONICAL) ,
data files "generlized.Dat" and "conical.Dat"
(data for the generalized finite elements, like in CLASSICAL and CONICAL),
executable DIPOG-2.1-GUI
(graphical user interface to replace the executables of CLASSICAL and CONICAL)

RESULTS → result files "name.res" and "name.erg"
(produced by executables in CLASSICAL and CONICAL)
executable PLOT_DISPLAY
(produces two-dimensional graph of data on the screen, argument "name.res" and indices of modes the efficiencies of which are to be plotted)
executable PLOT_PS
(produces ps file of two-dimensional graph of data, argument "name.res" and indices of modes the efficiencies of which are to be plotted)
executable PLOT_MATLAB
(produces Matlab file of three-dimensional graph of data, works only for classical illumination, argument "name.res" and indices of modes the efficiencies of which are to be plotted)
executable PLOT_GNUPLOT
(produces gnuplot ps file of three-dimensional graph of data, works only for classical illumination, argument

	"name.res" and indices of modes the efficiencies of which are to be plotted)
MAKES	→ header MHEAD (note that MHEAD is to be adapted to your computer system before installation), executable MAKEHOME (for another user: produces new version of six subdirectories GEOMETRIES, CLASSICAL, CONICAL, and RESULTS together with all data and example files and links to executables) body of makefile "makefile_all" and more

There exist subdirectories with technical files:

grid_tri	→ programs and input files for grating and grid data
dpogtr	→ programs and input file for the FEM computation in the classical case
gdpogtr	→ programs and input file for the GFEM computation in the classical case
conical	→ programs and input file for the FEM computation in the conical case
conical2	→ programs and input files for the GFEM computation in the conical case
optim	→ programs and input files for the optimization with OPTIMIZATION in the directory OPTIM
dipog-2.1-gui	→ programs, object files, and input files for the graphical user interface program DIPOG-2.1-GUI in the directory GUI
refr_ind_data	→ data files for the refractive indices
results	→ plot programs

Possibly, there exist subdirectories to install necessary packages[5]:

dipog-1.3	→ necessary source files from previous version of DIPOG-1.3, in dipog-1.3/gsl/src: files to install libhur.a
tgui	→ necessary sources to create TGUI in the directory GEOMETRIES
gltools_tar	→ necessary source files for Fuhrmann's package GLTOOLS-2.4, this produces subdirectory gltools-2-4 during installation
triangle[6]	→ necessary source files for Shewchuk's package TRIANGLE-1.4

In case of a simultaneous use of the package, each user has its own home directory containing the six subdirectories GEOMETRIES, CLASSICAL, CONICAL, OPTIM, GUI, and RESULTS. The subdirectories contain the same example and data files as described above for the directories of the package. However, the executables are replaced by symbolic links to the executables of the package.

[5]If needed, change to the subdirectories and follow the instructions of the corresponding files README.txt. If not needed, delete the subdirectories.

[6]This subdirectory exists only during internal installation.

1.5 Environment variables

The following environment variables are mandatory:

- `LD_LIBRARY_PATH`: This search path for loading program libraries must contain the address of the solver `PARDISO` (cf. Section 1.3).
- `OMP_NUM_THREADS`: This is to be set to the number of CPUs which should be used for solving linear systems of equations by `PARDISO` (cf. Section 1.3).

Additionally, the following environment variables can be set:

- `ADD_STRIPS`: If this is set to "yes" and if an optimization in the class of "bridges composed of trapezoids under light in the EUV range", is performed, then, automatically, a strip beneath the bridge is included into the Fem domain (cf. Section 10.2.7).
- `BND_MESH_SIZE`: If this is set to a positive number and if an optimization is performed, then the bound for the mesh size of the Fem partition at level one is set to this number (cf. Section 10.2.1).
- `BND_n_LFEM`: If this is set to a positive integer, then, for the non-local boundary value condition, the number of discretization points in each interval of the uniform partition of the upper and lower boundary line is set to this value (cf. Section 6).
- `CHOOSE_PMETHOD`: If this is set to "yes", then, like in the p-method of the Fem algorithm with $p = n_{\text{DOF}} + 1$ and with elimination of interior degrees of freedom, the local trial functions are the solution of a $[3\,p] \times [3\,p]$ system of equations (cf. Section 6).
- `COND_NMB_IT`: If this is set to a positive integer, if the environment variable `GET_COND_NUMB` is set, and if one of the executables `GFEM`, `GFEM_FULLINFO`, or `GFEM_PLOT` from the directory `DIPOG-2.1/CLASSICAL` is used, then this integer will be chosen as the number of iterations to improve the minimal eigenvalue in, the estimate of the condition number of the linear system of equations. The standard value is 10.
- `EFF_PLO`: If this is set to "yes", then the efficiencies for grating corresponding to the approximate solution are plotted and compared to the values prescribed in the objective functional.
- `EFFRES`: If this is set, then the efficiencies for the grating corresponding to the optimal solution are added to the result file "name.res".
- `EPS_OUT_VAL`: On screen and in result files only those numbers (efficiencies, energies, Rayleigh coefficients, phase shifts) are printed which are greater than $5 \cdot 10^{-9}\%$. However, this threshold $5 \cdot 10^{-9}$ can be changed to any positive number setting the environment variable `EPS_OUT_VAL`. Moreover, any value independent of its size is printed if `EPS_OUT_VAL` equals minus one.
- `EUV_SWA_90`: Performing an optimization in the class of EUV bridges (i.e. integer parameter i_geom_param[1]=6), the user can restrict the search to bridges with sidewall angle less or equal to 90° by setting `EUV_SWA_90` to "yes". This changes the meaning of the parameters (cf. Section 10.2.7).
- `GET_COND_NUMB`: If this is set, then the executables `GFEM`, `GFEM_FULLINFO`, and

GFEM_PLOT in the directory DIPOG-2.1/CLASSICAL will print estimates for the condition numbers instead of the memory requirements for the solver.

MIN_ANG_TRI: If this is set to a positive number and if no geometry input file "name.inp" is used, then the lower bound for the minimal angle of the triangles of the FEM partition is set to this number.

NMB_OF_DATA: Performing an optimization, the objective functional is allowed to depend on 999 values of efficiencies/phase shifts/energies. If this number is not sufficient, then the user can enlarge it setting the environment variable to the required number (cf. Section 10.1.3).

STE_DIF_FOR: If this is set to a positive number, then the standard step size $2 \cdot 10^{-7}$ in the difference formula for derivatives w.r.t. the parameters of the multi-layer system is changed to this value (cf. Section 10.2.7).

TMPDIR: If this is set to the name of an existing directory, then the temporary directory, used for auxiliary files in the programs, is changed from the directory prescribed during installation to the new one with the given name.

WI_GRA_LAY: Suppose an optimization of an EUV bridge without the sidewall angle restriction (i.e. with EUV_SWA_90="no") is required, where the bridge is located strictly in the middle of a period and where no additional layer is generated beside the bridge. If the electro-magnetic field changes fastly in the vicinity of the interface between cover material and grating, then a grading of the FEM grid towards this interface can improve the approximation essentially. This grading can be enforced by introducing a small "layer" beside the interface filled with cover material. The diameters of the triangles of the FEM grid will change smoothly from the maximal value to the width of this small layer. The width of this small layer is the number defined by the environment variable WI_GRA_LAY.

2 Diffraction Problems for Gratings

2.1 The classical TE problem

Consider an ideal optical grating (cf. the cross section in Figure 1). We choose the coordinate system such that the z-axis shows in the direction of the grooves and that the y-axis is orthogonal to the plane of the grooves. The width of the grooves in x direction is the periodicity d of the grating. The refractive index of the cover material is $\mathbf{n}^+$, that of the substrate under the grating surface structure $\mathbf{n}^-$. The grating part consists of several materials with indices $\mathbf{n}_i$. Above and below the grating structure there may exist some coated layers with different refractive index (cf. the indices $\mathbf{n}^{uc}$ and $\mathbf{n}^{lc}$ for one upper and one lower coating layer in Figure 1). We suppose that a plane wave is incident from above with a direction located in the $x-y$ plane (i.e. in the plane perpendicular to the grooves) and under the incident angle θ. The wave length of the light in air is λ and we consider the case of TE polarization where the electric field vector is parallel to the grooves, i.e. it shows in the z direction. Hence, if μ_0 is the magnetic permeability of vacuum and c the speed of light, then the transverse z coordinate of the electric field is given as

$$\begin{aligned}
\mathcal{E}_z^{incident}(x,y,z,t) &= E_z^{incident}(x,y,z)\ \exp(-\mathbf{i}\omega t)\,, \qquad \omega = \frac{2\pi c}{\lambda}\,, \\
E_z^{incident}(x,y,z) &= \frac{1}{\sqrt{\mathbf{n}^+}}\exp\big(\mathbf{i}k^+\sin\theta\, x - \mathbf{i}k^+\cos\theta\, y\big)\,, \quad k^+ = \omega\sqrt{\mu_0\varepsilon_0}\,\mathbf{n}^+.
\end{aligned} \tag{2.1}$$

The light is diffracted by the grating structure. Beside some evanescent part the diffracted light splits into a finite number of reflected and transmitted TE polarized plane wave modes, the propagation directions of which are independent of the grating geometry and the grating materials. The problem is to determine the amplitude and the phase of the reflected and transmitted modes. Note that the normalization factor $1/\sqrt{\mathbf{n}^+}$ in the second line of (2.1) has been introduced to obtain an incident light wave with a fixed intensity (length of Poynting vector) independent of the cover material. If the cover material is air, then $\mathbf{n}^+ = 1$ and the wave is normalized such that the amplitude of the electric field vector is of unit length. Of course the values of the efficiencies and phase shifts of the reflected and transmitted plane wave modes are independent of this normalization factor. The subsequent Rayleigh coefficients, however, depend on this scaling.

Using Maxwell's equations, it can be shown that the transverse component E_z satisfies the scalar Helmholtz equation $\{\triangle + k^2\}E_z = 0$ in any domain of the cross section plane with constant material as well as some transmission conditions on the interfaces between materials of different refractive indices. The wave number k is equal to ω/c times the refractive index of the material. Thus we can determine E_z by the standard method for elliptic differential equations by the FEM. Using the periodicity of the problem and standard coupling techniques with the boundary element method, the domain of numerical computation can be reduced to a rectangle Ω (cf. Figure 1). This covers one period of the grating and is bounded by the horizontal lines $\Gamma^\pm$ located inside the last upper and first lower coating layer (counted from above to below) resp. in the cover material and substrate material for gratings without coatings.[7]

Above resp. beneath the grating structure (including all the layers) the component E_z admits an expansion into the Rayleigh series of the form

$$E_z(x,y) = \sum_{n=-\infty}^{\infty} A_n^+ \exp\big(+\mathbf{i}\beta_n^+ y\big)\exp(\mathbf{i}\alpha_n x) + A_0^{inc}\exp\big(-\mathbf{i}\beta_0^+ y\big)\exp(\mathbf{i}\alpha x)\ , \tag{2.2}$$

$$E_z(x,y) = \sum_{n=-\infty}^{\infty} A_n^- \exp\big(-\mathbf{i}\beta_n^- y\big)\exp(\mathbf{i}\alpha_n x)\ , \tag{2.3}$$

$$\beta_n^\pm = \sqrt{[k^\pm]^2 - [\alpha_n]^2}\,, \quad k^\pm = \frac{\omega\mathbf{n}^\pm}{c}\,, \quad A_0^{inc} = \frac{1}{\sqrt{\mathbf{n}^+}}$$

$$\alpha = k^+\sin\theta, \qquad \alpha_n = k^+\sin\theta + \frac{2\pi}{d}n\ .$$

Here d is the period of the grating and the complex constants $A_n^\pm$ are the so-called Rayleigh

[7]For technical reasons in the FEM code, it is important to have the same material on both sides of the boundary lines $\Gamma^\pm$.

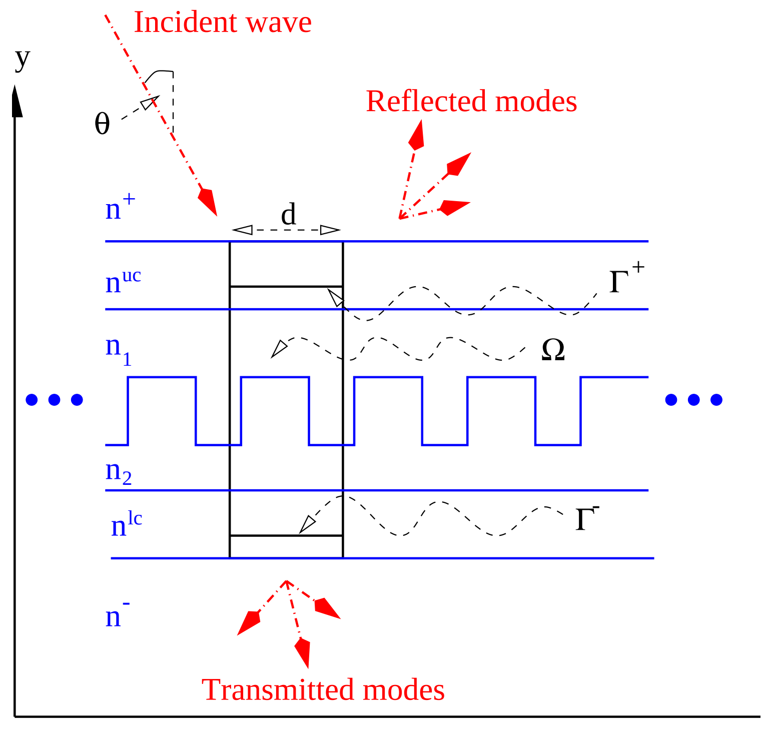


Figure 1: Cross section of grating.

coefficients. The interesting Rayleigh coefficients are those with $n \in \mathcal{U}^{\pm}$,

$$\mathcal{U}^{\pm} = \begin{cases} \left\{ n \in \mathbb{Z} : |\alpha_n| < k^{\pm} \right\} & \text{if } \Im\mathrm{m}\, k^{\pm} = 0 \\ \emptyset & \text{if } \Im\mathrm{m}\, k^{\pm} > 0 \end{cases} .$$

Indeed, these coefficients $A_n^{\pm}$ describe magnitude and phase shift of the propagating plane waves. More precisely, the modulus $|A_n^{\pm}|$ is the amplitude of the nth reflected resp. transmitted wave mode and $\arg[A_n^{\pm}/|A_n^{\pm}|]$ the phase shift. The terms with $n \notin \mathcal{U}^{\pm}$ lead to evanescent waves, only. The optical efficiencies of the grating are defined by

$$e_n^{\pm} = \frac{\beta_n^{\pm}}{\beta_0^{+}} \frac{|A_n^{\pm}|^2}{|A_0^{inc}|^2}, \quad (n, \pm) \in \left\{ (n,+) : \ n \in \mathcal{U}^{+} \right\} \cup \left\{ (n,-) : \ n \in \mathcal{U}^{-} \right\}, \tag{2.4}$$

which is the ratio of energy of the incident wave entailed to the nth propagating mode. Note that these efficiencies of propagating modes exist for non-absorbing materials, i.e. for $\Im\mathrm{m}\, k^{\pm} = 0$. If the transverse component E_z has been computed approximately, then the Rayleigh coefficients can be obtained by a discretized Fourier series expansion applied to the FEM solution restricted to $\Gamma^{\pm}$ (cf. (2.2) and (2.3)). Formula (2.4) yields the efficiencies.

2.2 The classical TM problem

The case of TM polarization is quite similar to TE. Indeed, this time the vector of the magnetic field $\mathcal{H}$ shows in the direction of the grooves, i.e. in the direction of the z axis.

Analogously to formula (2.1) given in the last subsection for the incident electric field, we get

$$\begin{aligned}\mathcal{H}_z^{incident}(x,y,z,t) &= H_z^{incident}(x,y,z)\ \exp(-\mathbf{i}\omega t)\,, \qquad (2.5)\\ H_z^{incident}(x,y,z) &= \frac{\sqrt{\varepsilon_0}\sqrt{\mathbf{n}^+}}{\sqrt{\mu_0}}\exp\left(\mathbf{i}k^+\sin\theta\, x - \mathbf{i}k^+\cos\theta\, y\right), \quad k^+ = \omega\sqrt{\mu_0\varepsilon_0}\,\mathbf{n}^+,\end{aligned}$$

for the z component of the incident magnetic field $\mathcal{H}_z^{incident}$. Note that the additional factor $\sqrt{\varepsilon_0}\sqrt{\mathbf{n}^+}/\sqrt{\mu_0}$ in the definition of $H_z^{incident}$ guarantees that the incident light wave has a fixed intensity (length of Poynting vector) independent of the cover material. If the cover material is air, then the wave is normalized such that the amplitude of the electric field vector is of unit length. Like in the TE case, the values of the efficiencies and phase shifts of the reflected and transmitted plane wave modes are independent of this normalization factor. Only the subsequent Rayleigh coefficients depend on this scaling.

The z component of the complete field H_z satisfies the Helmholtz equation $\{\triangle+k^2\}H_z = 0$ in any domain of the cross section plane with constant materials. However, the transmission conditions on the interfaces are different. We can solve the transmission problem of the Helmholtz equation by FEM. Again we have a finite number of transmitted and reflected modes and the Rayleigh expansions hold for E_z replaced by H_z. More precisely, the Rayleigh coefficients are the $B_n^\pm$ of the expansions

$$\frac{\sqrt{\mu_0}}{\sqrt{\varepsilon_0}}H_z(x,y) = \sum_{n=-\infty}^{\infty} B_n^+\exp\left(+\mathbf{i}\beta_n^+ y\right)\exp\left(\mathbf{i}\alpha_n x\right) + B_0^{inc}\exp\left(-\mathbf{i}\beta_0^+ y\right)\exp\left(\mathbf{i}\alpha x\right)\ , \quad (2.6)$$

$$\frac{\sqrt{\mu_0}}{\sqrt{\varepsilon_0}}H_z(x,y) = \sum_{n=-\infty}^{\infty} B_n^-\exp\left(-\mathbf{i}\beta_n^- y\right)\exp\left(\mathbf{i}\alpha_n x\right)\ , \quad (2.7)$$

$$B_0^{inc} = \sqrt{\mathbf{n}^+}\ .$$

The objective is to compute the Rayleigh coefficients. They result from the FEM solution of the new transmission problems and from the discretization of the Fourier series expansion (2.6) and (2.7). The efficiencies (cf. (2.4)) are computed by

$$e_n^\pm = \frac{\beta_n^\pm}{\beta_0^+}\frac{[k^+]^2}{[k^\pm]^2}\frac{|B_n^\pm|^2}{|B_0^{inc}|^2}, \quad (n,\pm)\in\Big\{(n,+):\ n\in\mathcal{U}^+\Big\}\cup\Big\{(n,-):\ n\in\mathcal{U}^-\Big\}. \quad (2.8)$$

Finally, we note that the case of an incident wave propagating in a direction of the $x-y$ plane together with an arbitrary polarization is the superposition of TE and TM polarization.

2.3 Conical problems

The essential difference between the classical diffraction of the last two subsections and the conical one is that the direction of the incident light wave is oblique, i.e. it is not restricted to the $x-y$ plane. Whereas in the classical case the directions of the finitely many reflected and transmitted plane wave modes remain located in the $x-y$ plane, now they are located on a cone in the $x-y-z$ space. The FEM approach is analogous to the classical case. However, instead of a transmission problem for a scalar Helmholtz equation, Maxwell's system reduces to a coupled system of two scalar Helmholtz equations for the two transverse components E_z and H_z of the electric and magnetic field. Consequently, we have two Rayleigh expansions and two sequences of Rayleigh coefficients.

More precisely, skipping the time harmonic factor, we have an incident wave of the form

$$\begin{aligned} E^{incident}(x,y,z) &= E^{inc}\exp\Big(\mathbf{i}[\alpha x - \beta y + \gamma z]\Big)\,, \\ H^{incident}(x,y,z) &= H^{inc}\exp\Big(\mathbf{i}[\alpha x - \beta y + \gamma z]\Big)\,. \end{aligned}$$

with constant vectors E^{inc} and H^{inc} and a wave vector $\vec{k} = (\alpha, -\beta, \gamma)$ such that the wave number k^+ is the modulus of $\vec{k}$ and the direction of the incoming plane wave is $\vec{k}/k^+$. Here k^+ is the same wave number as in the classical TE and TM case. The direction $\vec{k}/k^+$ must be prescribed by the user of `DIPOG-2.1`. This can be characterized by two parameters, namely by the angles θ_i and ϕ_i which are the spherical coordinates of (α, β, γ). We emphasize that θ_i and ϕ_i are the spherical coordinates of (α, β, γ) and not those of the normalized wave vector $\vec{k}/k^+ = (\alpha, -\beta, \gamma)$.[8] Contrary to this, the angles θ and ϕ of the reflected and transmitted plane wave modes are exactly the spherical coordinates of the normalized wave vectors. Unfortunately, this traditional notation is a little bit confusing.

Either we use the spherical coordinate system with the $x-y$ plane as basis plane (xy system) or the spherical coordinates based on the $x-z$ plane (xz system). In the xy system we define the direction $D_{xy} = (\alpha, \beta, \gamma)$ as (cf. Figure 2):

$$D_{xy} = \Big(\sin\theta_{xy}\cos\phi_{xy}\ ,\ \cos\theta_{xy}\cos\phi_{xy}\ ,\ \sin\phi_{xy}\Big)$$

Here $\theta_{xy} \in (-90°, 90°)$ is the angle of inclination of the plane, containing the direction D_{xy} and the z axis, from the $y-z$ plane. Angle $\phi_{xy} \in (-90°, 90°)$ is the angle of direction D_{xy} inside this inclined plane, i.e. the angle between D_{xy} and the orthogonal projection of D_{xy} to the $x-y$ plane.

For the xz system the direction $D_{xz} = (\alpha, \beta, \gamma)$ is given by (cf. Figure 3):

$$D_{xz} = \Big(\sin\theta_{xz}\cos\phi_{xz}\ ,\ \cos\theta_{xz}\ ,\ \sin\theta_{xz}\sin\phi_{xz}\Big)$$

Here ϕ_{xz} is the angle of inclination of the plane, containing the direction D_{xz} and the y axis, from the $x-y$ plane. Angle $\theta_{xz} \in [0, 90°)$ is the angle of direction D_{xz} inside this inclined plane, i.e. the angle between D_{xz} and the y-axis. To change between the xy system and the xz system the following formulae are useful

$$\begin{aligned} \phi_{xy} &= \arcsin\Big(\sin\theta_{xz}\sin\phi_{xz}\Big)\,, \quad \theta_{xy} = \arcsin\left(\frac{\sin\theta_{xz}\cos\phi_{xz}}{\sqrt{1-\sin^2\theta_{xz}\sin^2\phi_{xz}}}\right)\,, \\ \theta_{xz} &= \arccos\Big(\cos\theta_{xy}\cos\phi_{xy}\Big)\,, \end{aligned}$$

[8] In other words $\pi - \theta_i$ and ϕ_i are the spherical coordinates of $\vec{k}/k^+$, and θ_i is not the angle enclosed by $\vec{k}/k^+$ and the positive y axis but the angle enclosed by $\vec{k}/k^+$ and the negative y axis.

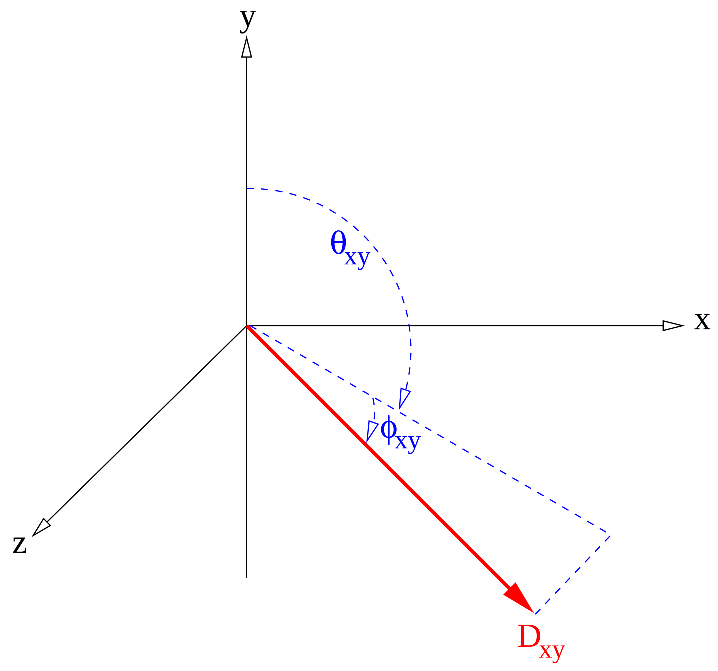


Figure 2: Coordinate system based on $x - y$ plane.

$$\phi_{xz} \;=\; \begin{cases} \arcsin\left(\dfrac{\sin\phi_{xy}}{\sqrt{1-\cos^2\theta_{xy}\cos^2\phi_{xy}}}\right) & \text{if } \theta_{xy} > 0 \\ \pm\pi - \arcsin\left(\dfrac{\sin\phi_{xy}}{\sqrt{1-\cos^2\theta_{xy}\cos^2\phi_{xy}}}\right) & \text{else .} \end{cases}$$

Though the user can choose his favourite spherical coordinate system for the input of the direction of incidence, the output of the directions for the reflected and transmitted modes are presented in the xz system.

Clearly, the fields E^{inc} and H^{inc} must be orthogonal. Moreover, the two vectors are uniquely determined by the normalization condition, by Maxwell's equation, and by the polarization type prescribing the polarization direction. Here the normalization condition

$$\mathbf{n}^+[E_z^{inc}]^2 + \frac{1}{\mathbf{n}^+}[B_z^{inc}]^2 \;=\; \frac{[k^+]^2-\gamma^2}{[k^+]^2}, \quad B_z^{inc} \;=\; \frac{\sqrt{\mu_0}}{\sqrt{\varepsilon_0}}H_z^{inc}$$

means that the incident light wave has a fixed intensity (length of Poynting vector) independent of the cover material. If the cover material is air, then the wave is normalized such that the amplitude of the electric field vector is of unit length. The values of the efficiencies and phase shifts of the reflected and transmitted plane wave modes are not effected by the normalization factor. Only the subsequent Rayleigh coefficients depend on this scaling.

The polarization type must be prescribed by the user of `DIPOG`-2.1. There are three possibilities. The first is to choose TE polarization with the electric field vector E^{inc} pointing in the direction perpendicular to the wave vector (incidence direction) and to the y axis. The second is TM polarization with the magnetic field vector H^{inc} pointing in the direction perpendicular to the wave vector (incidence direction) and to the y axis. Note that the direction perpendicular to the wave vector and to the y axis is, by definition, the z axis if wave vector and y axis should be collinear. The third choice is to prescribe the angle ψ (cf. Figure 4) enclosed by the x axis and by the projection of the electric field vector E^{inc} to the $x - z$ plane.

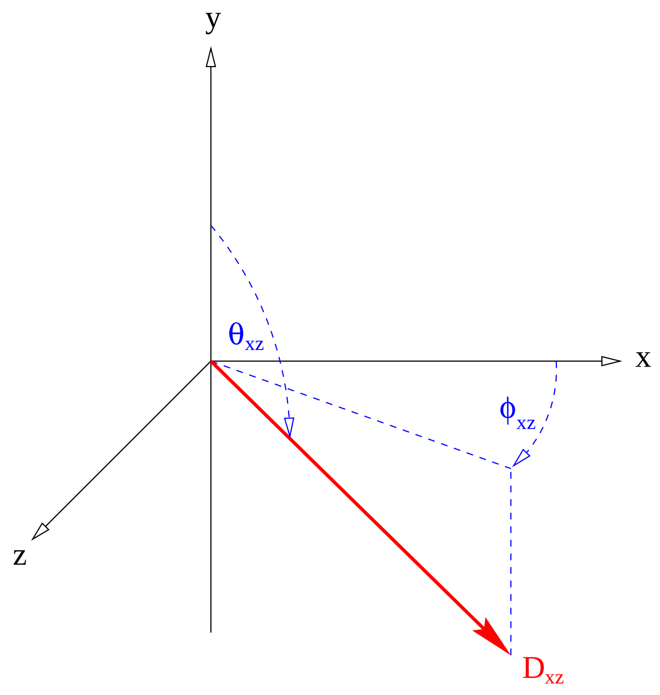


Figure 3: Coordinate system based on $x - z$ plane.

Fixing the incident field, the resulting total field is determined and can be computed by FEM. It remains to describe the output data of `DIPOG`-2.1. The Rayleigh expansions above resp. below the grating take the form

$$\begin{aligned} E(x,y,z) &= E^{inc} \exp\Big(\mathbf{i}[\alpha x - \beta y + \gamma z]\Big) + \sum_{n\in\mathbb{Z}} \vec{A}_n^+ \exp\Big(\mathbf{i}[\alpha_n x + \beta_n^+ y + \gamma z]\Big) , \\ H(x,y,z) &= H^{inc} \exp\Big(\mathbf{i}[\alpha x - \beta y + \gamma z]\Big) + \sum_{n\in\mathbb{Z}} \vec{C}_n^+ \exp\Big(\mathbf{i}[\alpha_n x + \beta_n^+ y + \gamma z]\Big) , \\ & \qquad \alpha_n = \alpha + \frac{2\pi}{d} n, \ \beta_n^\pm = \sqrt{[k^\pm]^2 - [\alpha_n]^2 - \gamma^2} \ , \ \Re\mathrm{e}\, \beta_n^\pm > 0 \ , \ \Im\mathrm{m}\, \beta_n^\pm \geq 0 \end{aligned}$$

resp.

$$\begin{aligned} E(x,y,z) &= \sum_{n\in\mathbb{Z}} \vec{A}_n^- \exp\Big(\mathbf{i}[\alpha_n x - \beta_n^- y + \gamma z]\Big) , \\ H(x,y,z) &= \sum_{n\in\mathbb{Z}} \vec{C}_n^- \exp\Big(\mathbf{i}[\alpha_n x - \beta_n^- y + \gamma z]\Big) . \end{aligned}$$

Now there are three variants of output data. The first computes the third components, i.e. the z components of the Rayleigh coefficients

$$p_n^\pm = [\vec{A}_n^\pm]_z \ , \quad q_n^\pm = [\vec{B}_n^\pm]_z \ , \quad \vec{B}_n^\pm = \frac{\sqrt{\mu_0}}{\sqrt{\varepsilon_0}} \vec{C}_n^\pm$$

and the efficiencies

$$\begin{aligned} e_n^+ &= \frac{\beta_n^+}{\beta} \frac{[k^+]^2}{[k^+]^2 - \gamma^2} \left[\mathbf{n}^+ |p_n^+|^2 + \frac{1}{\mathbf{n}^+} |q_n^+|^2 \right] , \qquad (2.9) \\ e_n^- &= \frac{\beta_n^-}{\beta} \frac{1}{[k^-]^2 - \gamma^2} \left[\mathbf{n}^+ [k^-]^2 |p_n^-|^2 + \frac{1}{\mathbf{n}^+} [k^+]^2 |q_n^-|^2 \right] \end{aligned}$$

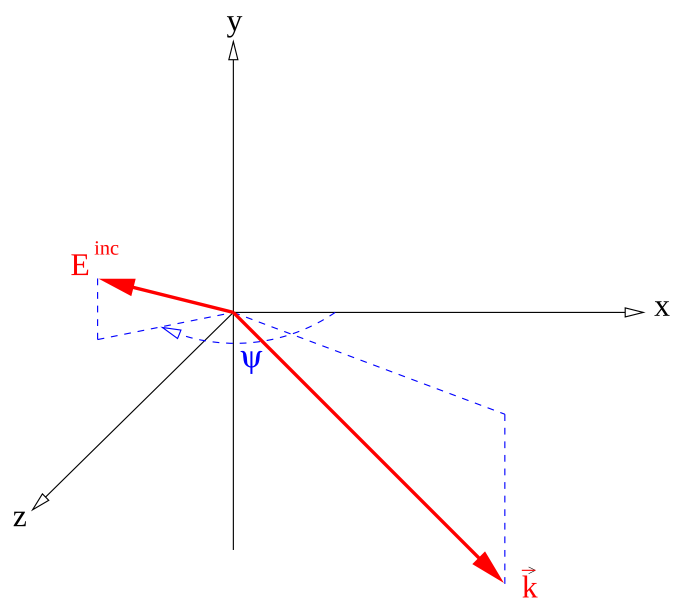


Figure 4: Coordinate system based on $x - z$ plane.

of the nth reflected resp. transmitted wave mode. The second output variant computes the TE and TM part of the total wave, i.e. if $\mathbf{s}_n^\pm$ stands for the direction perpendicular to the y axis and to the direction of propagation of the nth reflected resp. transmitted plane wave mode ($\mathbf{s}_n^\pm = (\alpha_n, \pm\beta_n^\pm, \gamma) \times (0,1,0)/|(\alpha_n, \pm\beta_n^\pm, \gamma) \times (0,1,0)|$), then the output coefficients are the scalar products

$$\begin{aligned}
\left\langle \vec{A}_n^\pm, \mathbf{s}_n^\pm \right\rangle &= \frac{ap_n^\pm + bc\,q_n^\pm/\mathbf{n}^\pm}{(1-c^2)\sqrt{a^2+c^2}}\,, \\
\left\langle \vec{B}_n^\pm, \mathbf{s}_n^\pm \right\rangle &= \frac{aq_n^\pm - bc\,p_n^\pm \mathbf{n}^\pm}{(1-c^2)\sqrt{a^2+c^2}}\,, \quad (a,b,c) = \frac{(\alpha_n, \pm\beta_n^\pm, \gamma)}{\sqrt{\alpha_n^2 + [\beta_n^\pm]^2 + \gamma^2}}
\end{aligned}$$

The efficiencies of the second output are the total efficiencies $e_n^\pm$ of (2.9) and the efficiencies corresponding to the TE and TM parts

$$\frac{\beta_n^\pm}{\beta} \left| \left\langle \vec{A}_n^\pm, \mathbf{s}_n^\pm \right\rangle \right|^2 \mathbf{n}^+\,, \quad \frac{\beta_n^\pm}{\beta} \frac{\mathbf{n}^+}{[\mathbf{n}^\pm]^2} \left| \left\langle \vec{B}_n^\pm, \mathbf{s}_n^\pm \right\rangle \right|^2\,,$$

i.e. the efficiencies of the projection of the nth reflected resp. transmitted wave mode to the component with electric resp. magnetic field polarized in $\mathbf{s}_n^\pm$ direction. Finally, the third variant computes the S- and P-parts of the electric field, i.e. the components of the Jones vector. If $\mathbf{s}_n^\pm$ is defined as above and if $\mathbf{p}_n^\pm$ is the direction orthogonal to $\mathbf{s}_n^\pm$ and the direction of propagation of the nth reflected resp. transmitted plane wave mode ($\mathbf{p}_n^\pm = (\alpha_n, \pm\beta_n^\pm, \gamma) \times \mathbf{s}_n^\pm/|(\alpha_n, \pm\beta_n^\pm, \gamma) \times \mathbf{s}_n^\pm|$), then the S- and P-parts of the Rayleigh coefficients are

$$\left\langle \vec{A}_n^\pm, \mathbf{s}_n^\pm \right\rangle = \frac{ap_n^\pm + bc\,q_n^\pm/\mathbf{n}^\pm}{(1-c^2)\sqrt{a^2+c^2}}\,,$$

$$\left\langle \vec{A}_n^{\pm}, \mathbf{p}_n^{\pm} \right\rangle \;=\; -\frac{1}{\mathbf{n}^{\pm}} \left\langle \vec{B}_n^{\pm}, \mathbf{s}_n^{\pm} \right\rangle \;=\; \frac{bc\, p_n^{\pm} - a\, q_n^{\pm}/\mathbf{n}^{\pm}}{(1-c^2)\sqrt{a^2+c^2}}\,.$$

The efficiencies of the third output are the total efficiencies $e_n^{\pm}$ of (2.9) and the efficiencies corresponding to the S- and P-parts, i.e. the efficiencies of the projection of the nth reflected resp. transmitted wave mode to the component with electric field polarized in $\mathbf{s}_n^{\pm}$ resp. $\mathbf{p}_n^{\pm}$ direction

$$\frac{\beta_n^{\pm}}{\beta} \left| \left\langle \vec{A}_n^{\pm}, \mathbf{s}_n^{\pm} \right\rangle \right|^2 \mathbf{n}^{+}\,, \quad \frac{\beta_n^{\pm}}{\beta} \left| \left\langle \vec{A}_n^{\pm}, \mathbf{p}_n^{\pm} \right\rangle \right|^2 \mathbf{n}^{+}\,.$$

2.4 References

For more details, see the following publications and the references therein:

- G. Bao, D.C. Dobson, and J.A. Cox: Mathematical studies in rigorous grating theory, *J. Opt. Soc. Amer. A* **12**, pp. 1029–1042 (1995).
- J. Elschner and G. Schmidt: Diffraction in periodic structures and optimal design of binary gratings I: Direct problems and gradient formulas, *Math. Meth. Appl. Sci.* **21**, pp. 1297–1342 (1998).
- J. Elschner and G. Schmidt: The numerical solution of optimal design problems for binary gratings, *J. Comput. Physics* **146**, pp. 603–626 (1998).
- J. Elschner, R. Hinder and G. Schmidt: Finite element solution of conical diffraction problems, *Adv. Comput. Math.* **16**, pp. 139–156 (2002).
- J. Elschner, R. Hinder and G. Schmidt: Direct and inverse problems for Diffractive Structures- Optimization of binary gratings, In: W. Jäger, H.J. Krebs (eds.), *Mathematics, key technology for the future: joint projects between universities and industry*, Springer Verlag Berlin Heidelberg, 2003, pp. 293–304.
- R. Petit (ed.): *Electromagnetic Theory of Gratings*, Springer, Berlin, 1980.
- H.P. Urbach: Convergence of the Galerkin method for two-dimensional electromagnetic problems, *SIAM J. Numer. Anal.* **28**, pp. 697–710 (1991).

For generalized FEM methods applied to the Helmholtz equation (modified and described in the subsequent Sections 6 and 7), we refer to:

- I. Babuška, F. Ihlenburg, E. Paik, and S. Sauter: A generalized finite element method for solving the Helmholtz equation in two dimensions with minimal pollution, *Comp. Methods Appl. Mech. Eng.* **128**, pp. 325–359 (1995).
- O. Cessenat and B. Depres: Application of an ultra weak variational formulation of elliptic PDEs to the two-dimensional Helmholtz problem, *SIAM J. Numer. Anal.* **35**, 255-299 (1998).
- F. Ihlenburg: *Finite element analysis of acoustic scattering*, Springer Verlag New-York Berlin Heidelberg, Applied Mathematical Sciences **132**, 1998.
- J.M. Melenk and I. Babuška: The partition of unity method: Basic theory and applications, *Comp. Methods Appl. Mech. Eng.* **139**, pp. 289–314 (1998).

For the solver of the linear system of equations, we refer to:

- O. Schenk, K. Gärtner, and W. Fichtner: Efficient Sparse LU Factorization with Left-Right Looking Strategy on Shared Memory Multiprocessors, *BIT*, Vol.40, 158-176 (2000).
- O. Schenk and K. Gärtner: Solving unsymmetric sparse systems of linear equations with `PARDISO`, *Journal of Future Generation Computer Systems* **20**, pp. 475–487, 2004.
- O. Schenk and K. Grtner: On fast factorization pivoting methods for symmetric indefinite systems, *Elec. Trans. Numer. Anal.* **23**, pp. 158–179, 2006.

3 Geometry Input

3.1 Geometrical data in input file “name.dat”

Computation starts with the change of the working directory to directory `CLASSICAL` (for classical diffraction) or to `CONICAL` (conical diffraction) and by calling an executable (e.g. `FEM` or `GFEM`) followed by the data file “name.dat” as argument of the executable. Here “name.dat” contains all information on the grating and the light.

```
cd DIPOG-2.1/CLASSICAL
FEM name.dat
```

or

```
cd DIPOG-2.1/CONICAL
GFEM name.dat
```

On the screen there will appear the output data of the computation and the name of an additional output file, where the output data is stored.

Mainly, the geometrical information of the input data in “name.dat” is fixed by the lines:

```
# Length factor of additional shift of grating geometry.
#       This is shift into the x-direction.
#       This is length of shift relative to period.
   0.

# Stretching factor for grating in y-direction:
   1.

# Length of additional shift of grating in micro meter.
#       This is shift in y-direction.
   0.

# Period of grating in micro meter:
   1.

# Grating data:
   name1
```

Here “name1” refers either to a file “name1.inp” with geometrical data located in the subdirectory `GEOMETRIES` or to some special code words to fix the geometry of the grating. We describe how to get the file “name1.inp” in point 3.2 and the alternative code words in the subsequent point 3.5 of this section.

As mentioned above, the computation starts in the directory "`CLASSICAL`" and is based upon a geometry input file "`GEOMETRIES`/name1.inp" indicated in "name.dat". However, if the code is started from a directory different from "`CLASSICAL`" or if the geometry data file is located in a different directory with the path "path1", then the file is to be specified by adding its path in "name.dat" as:

```
# Grating data.
  path1/name1
```

In particular, for a geometry input file in the current working directory use:

```
# Grating data.
  ./name1
```

Note that the geometry data in "name1.inp" should be given relative to the period which is specified in the data file "name.dat" of directory `CLASSICAL` resp. `CONICAL`. All data of "name1.inp" will later be multiplied by the given length of period (e.g. by 1 μm).

Additional geometrical information, fixed in file "name.dat", concerns the coated layers. In principle, the grating part is a rectangular domain (cf. Ω in figure 1). Above and below this part we can add a few number of coated layers in form of strips parallel to the upper and lower side of the rectangle. The numbers of these layers together with the corresponding thickness is given in special lines of "name.dat". Explanations of the lines in "name.dat" can be found directly in the neighbouring comment lines starting with symbol "#".

3.2 How to get an input file "name1.inp"?

The elementary way to create "name1.inp" is the following. Change to subdirectory `GEOMETRIES`. Copy an existing file like e.g. "example.inp" (cf. the enclosed file in 12.1), and change its name into e.g. "name1.inp".

```
cd DIPOG-2.1/GEOMETRIES
cp example.inp name1.inp
```

Change "name1.inp" in your editor (emacs,vi?) according to your requirements. You will find the necessary information as comments in the file "name1.inp". Indeed, each line beginning with "#" is a comment. For example, the number n_{mat} of different materials is fixed in "name1.inp" by the input lines:

```
# Number of materials:
  n_mat
```

We emphasize that this number must include the two materials located immediately over and under the grating area since two rectangular layers from these adjacent regions are added to the area of FEM computation. Consequently, $n_{mat} \geq 2$, and $n_{mat} = 2$ holds if the grating structure is manufactured from the same two materials filling the regions adjacent to the grating structure. In the case of the grating in Figure 1, we have two materials with the refractive indices n_1 and n_2 inside the grating structure and two adjacent materials with indices n^{lc} and n^{uc}, i.e. $n_{mat} = 4$. Generally, the materials inside the grating structure are to be indicated in "name1.inp" by an index between 1 and n_{mat}. In particular, index 1

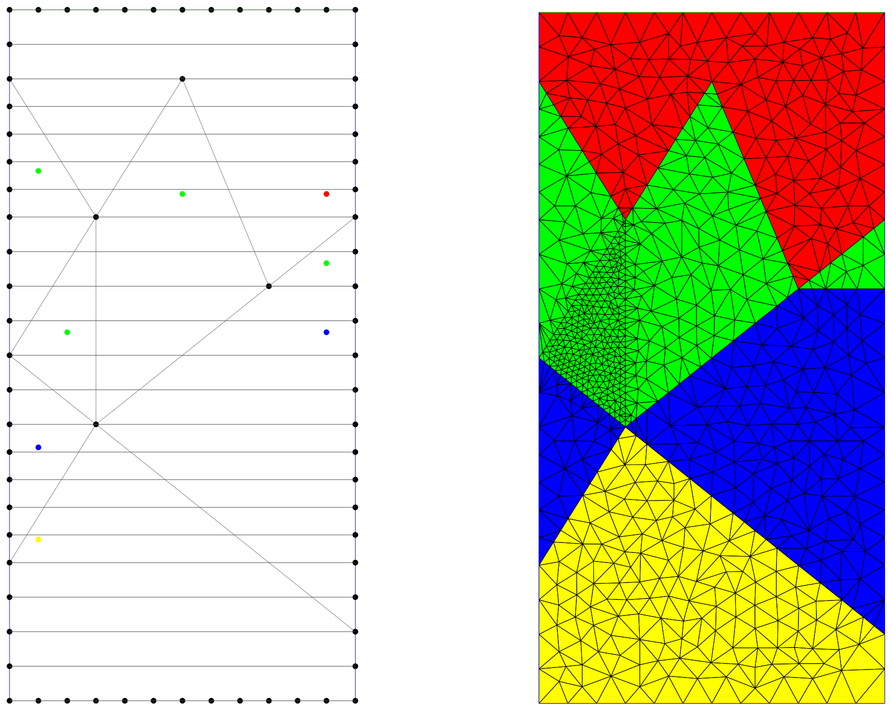


Figure 5: Pictures of grid produced by `SHOW`.

stands for the material immediately over the grating and n_{mat} for that immediately under it. Of course, the corresponding n_{mat} refractive indices are listed in the input file "name.dat" and not in "name1.inp". Finally, we remark that the file "name1.inp" contains its name "name1" without the tag ".inp". This name must include the complete path if the file is not located in the directory `GEOMETRIES`.

To check the geometry described by "name1.inp", enter the command:

```
SHOW name1.inp
```

You will see a first picture (cf. left picture in Figure 5) with the chosen points of a polygonal structure. After pressing Escape or Bar/Space you see a second picture (cf. right picture in Figure 5) with a coarse triangulation and with the different regions (later distinguished by different optical indices) in different colours. Press Escape or Bar/Space to end the check. If you enter

```
SHOW v name1.inp
```

then, additionally, an eps-file of the picture is produced.

Alternatively, to create "name1.inp", one can call the executable `GEN_INPUT` from the subdirectory `GEOMETRIES` and work interactively. Just enter the command:

```
GEN_INPUT
```

This program prompts you for everything needed. Nevertheless, we recommend the first way of copying and modifying an existing file.

The programs of `DIPOG-2.1` are based on a coupling of finite elements and boundary elements over the upper and lower boundary lines of the Fem domain. This coupling requires that the refractive indices of the materials on both sides of the boundary lines coincide. Therefore the Fem domain is extended by additional rectangular strips adjacent to the upper and lower boundary lines. The positive width (relative to the period) of these two strips is fixed in the "name1.inp" file, e.g. by the lines:

```
# Width of additional strip above and below:
  0.5
```

The material of the upper strip is the grating material of index one, that of the lower has index n_{mat}. In order to have the same material on both sides of the upper and lower boundary lines, the refractive index of the lowest upper coated layer resp. the cover material must be the same as that of the grating material with index one, and the refractive index of the highest lower coating layer resp. the substrate material must be the same as that of the grating material with index n_{mat}. Adding the widths of the upper and lower coated layers, the user must not forget about the rectangular strips included already in the grating structure.

The width of the additional rectangular strips adjacent to the upper and lower boundary lines can be chosen automatically by adding the width input zero in the "name1.inp" file. More precisely, adding a zero for the width, the width is set to

$$\min\{0.05\,,\ \text{upper bound of meshsize}\} \times \text{period} \tag{3.10}$$

which approximates zero for the meshsize tending to zero. The new rectangular strips are borrowed from the adjacent coated layer resp. from the substrate or cover material, i.e. the widths of the adjacent coated layers are reduced by the width of the strip. In other words, the automatic choice of the widths of the additional strips requires that the expression in (3.10) is less than the widths of the adjacent coated layers.

Now suppose that the width of the additional strips is chosen automatically and that there exist upper coated layers above the grating geometry fixed by the "name1.inp" file. The natural starting point of the additional upper strip is the point of the grating geometry with the highest y-coordinate which belongs to an area occupied by a material different from that of the adjacent upper coated layer. For technical reasons, the grating geometry without the additionally added layers must not contain a strip of the upper coating material above the natural starting point of the strip to be added automatically. Similarly, suppose that the width of the additional strips is chosen automatically and that there exist lower coated layers below the grating geometry. The natural starting point of the additional lower strip is the point of the grating geometry with the lowest y-coordinate which belongs to an area occupied by a material different from that of the adjacent lower coated layer. Again, for technical reasons, the grating geometry without the additionally added layers must not contain a strip of the lower coating material below the natural starting point.

If the coated layer adjacent to the boundary line is very thin, then the automatically added additional layer in the Fem domain is thin, and a huge number of small triangles appear in the triangulation. The resulting large number of degrees of freedom can be avoided in the case of classical Te polarization, where the coupling of finite elements and boundary elements does not require the same material on both sides of the boundary lines.

Avoiding an additional strip below resp. above the FEM domain requires two assumptions:

- The upper resp. lower polygonal boundary line of the FEM domain should form a horizontal straight line segment.[9]
- All corners with maximal resp. minimal y coordinate must have an x coordinate which is the product of period times a rational number k/l with $l < 1000$. [10]

To switch off the automatic generation of additional strips in the FEM domain, the user must work with a geometry input file "name1.inp" containing the lines:

```
# Width of additional strip above and below:
  no
```

Sometimes the additional strip is required below the grating structure, but it should be switched off above it. This can be indicated by

```
# Width of additional strip above and below:
  no_up 0.
```

Here the number following the no_up is the thickness of the additional layer beneath the grating. Similarly, an additional strip above and no strip below can be indicated by

```
# Width of additional strip above and below:
  no_lo 0.
```

If the automatic generation of additional strips is switched off, then the input counter of the materials n_{mat} as well as the list of refractive indices of the grating must not include the materials of the omitted additional strips.

As mentioned above, special gratings like echelle gratings, lamellar, trapezoidal, and simple profile gratings need not to be generated by an input file "name1.inp". Special code words will generate automatically hidden files of this type. However, in some situations the user might wish to change the automatically generated "name1.inp" files. He might wish to add small modifications to the geometry, or he wants to change the meshsize. To do this the user can create the otherwise hidden "name1.inp" files explicitly by the following executables.

If an input file for an echelle grating of type A is needed (right-angled triangle with hypotenuse parallel to the direction of the periodicity, cf. Figure 6), then this can be accomplished by calling the executable `GEN_ECHELLEA` from the subdirectory `GEOMETRIES`. More precisely, the command

```
GEN_ECHELLEA name R 0.3 0.03 0.04
```

creates the file "name.inp" of the desired echelle profile grating, with right blaze angle greater than 45°, with a depth (triangle height) of 0.3 times period of the grating and with coated layers of height 0.03 resp. 0.04 times period over the first resp. second part of the grating (measured in direction perpendicular to the echelle profile, height greater or equal to zero). If the input letter **R** is replaced by an **L**, then the left blaze angle is greater than

[9] If this formal assumption is not fulfilled, then a rectangular upper resp. lower coating strip does not make sense and, without coated layers, a thicker artificial strip can be added without problem

[10] Indeed, the corner points at the upper resp. lower boundary lines of the FEM domain must be part of a uniform grid. This is required by the fast boundary element discretization.

45 degrees. Moreover, if the input letter **R** is replaced by an **A** and the following input number 0.3 by 60., then the left blaze angle is 60°.

If an input file for an echelle grating of type B is needed (right-angled triangle with one of the legs parallel to the direction of the periodicity, cf. Figure 7), then this can be accomplished by calling the executable `GEN_ECHELLEB` from the subdirectory `GEOMETRIES`. More precisely, the command

```
GEN_ECHELLEB name 60. 0.05
```

creates the file "name.inp" of the desired echelle profile grating, with angle 60° (angle enclosed by hypotenuse and by the leg parallel to the period) and with a coated layer of height 0.05 times period of the grating (measured in direction perpendicular to echelle profile, height greater or equal to zero).

If an input file for a general echelle grating is needed, then this can be accomplished by calling the executable `GEN_ECHELLE` from the subdirectory `GEOMETRIES`. More precisely, the command

```
GEN_ECHELLE name A 120. L 30. 0.05 0.1
```

creates the file "name.inp" of the desired echelle profile grating, with an apex angle of 120°, with a left blaze angle of 30°, with a coated layer over the left blaze side of height 0.05 times length of period of the grating (measured in direction perpendicular to echelle profile, height greater or equal to zero), and with a coated layer over the right blaze side of height 0.1 times length of period of the grating (measured in direction perpendicular to echelle profile, height greater or equal to zero, must be zero if previous height is zero). Instead of the two inputs "A 120." and "L 30." one can choose also the inputs "R 110." for a right blaze angle of 110° or "D 0.4" for a depth of the grating equal to 0.4 times length of period of the grating. Any combination of two inputs of the types "A 120.", "L 30.", "R 110.", and "D 0.4" is accepted. However, the choice "A 120." and "D 0.2" is ambiguous. By definition it fixes an echelle grating with right blaze angle larger than the left. To get the flipped grating with left blaze angle larger than the right, the input should be "A 120." and "D -0.2". Figure 6 corresponds to a call of `GEN_ECHELLE` with the parameter arguments "A 90. D 0.3 0.03 0.04".

If an input file for a trapezoidal grating is needed (isosceles trapezoid with the basis parallel to the direction of the periodicity, cf. Figure 8), then this can be accomplished by calling the executable `GEN_TRAPEZOID` from the subdirectory `GEOMETRIES`. More precisely, the command

```
GEN_TRAPEZOID name 60. 0.6 3 0.2 0.1 0.1 0.05
```

creates the file "name.inp" of the desired trapezoidal profile grating, with angle 60° (angle enclosed by basis and the sides) with a basis of length 0.6 times period of the grating consisting of 3 material layers of heights 0.2 times period, 0.1 times period, and 0.1 times period, respectively, and with a coated layer of height 0.05 times period (greater or equal to zero).

If an input file for a grating with several trapezoids one beside the other is needed (symmetric trapezoids with the basis parallel to the direction of the periodicity), then this can be accomplished by calling the executable `GEN_MTRAPEZOID` from the subdirectory `GEOMETRIES`.

More precisely, the command

GEN_MTRAPEZOID name1 name2

creates the file "name1.inp" of the desired multitrapezoidal profile grating, with data taken from the input file "name2.INP". In particular, this contains lines with (all lengths relative to period, all angles in degrees)
– number m of layers in one bridge
– m heights of the layers (from above to below)
– m sidewall angles (from above to below)
– height at which lateral width is given
– number n of bridges
– n lateral widths of bridges
– n x-coordinates of midpoints of bridges.

If an input file for a lamellar grating is needed (rectangular grating consisting of several materials placed in rectangular subdomains, cf. Figure 10), then this can be accomplished by calling the executable GEN_LAMELLAR from the subdirectory GEOMETRIES. More precisely, if the file "GEOMETRIES/lamellar.INP" contains the numbers (each number in a separate line) 3, 4, 0.2, 0.6, -0.2, 1.0, 0., 0.5, 0.70, .0, 0.50, 0.900, .00, 0.500, and 0.900, then the command

GEN_LAMELLAR name lamellar.INP

creates the file "name.inp" of the desired lamellar profile grating, with 3 columns each divided into 4 rectangular layers, first column with x coordinate in $0 < x < 0.2$, second column with $0.2 < x < 0.6$, third column with $0.6 < x < 1$ (all coordinates are normalized with respect to the period: period corresponds to x=1), whole grating with y coordinate s.t. $-0.2 < y < 1.0$, first column: first layer with $-0.2 < y < 0.$, second with $0. < y < 0.5$, third with $0.5 < y < 0.7$ and fourth with $0.7 < y < 1.$, second column: first layer with $-0.2 < y < 0.0$, second with $0.0 < y < 0.50$, third with $0.50 < y < 0.90$ and fourth with $0.90 < y < 1.$, third column: first layer with $-0.2 < y < 0.00$, second with $0.00 < y < 0.500$, third with $0.500 < y < 0.900$ and fourth with $0.900 < y < 1.$.

If an input file for a simple layer grating is needed, then this can be accomplished by calling the executable GEN_LAMELLAR from the subdirectory GEOMETRIES. More precisely, if the file "GEOMETRIES/lamellar.INP" contains the numbers (each number in a separate line) 1, 1, 0.2, and 0.8, then the command

GEN_LAMELLAR name lamellar.INP

creates the file "name.inp" of the desired layer grating, with layer material s.t. the y coordinate satisfies $0.2 < y < 0.8$ (all coordinates are normalized with respect to the period: period corresponds to x=1).

If an input file for a grating with a polygonal profile line is needed (cf. Figure 11), then this can be accomplished by calling the executable GEN_POLYGON from the subdirectory GEOMETRIES. More precisely, if the file "GEOMETRIES/file1" contains the corner points of a polygonal profile line (in "GEOMETRIES/file1": in each line beginning without '#' there should be the x- and y-coordinate of one of the consecutive corner points, first point with

x-coordinate 0, last point with x-coordinate 1, same y-coordinate for first and last point, all x-coordinates between 0 and 1, at least two different y-coordinates, last line should be "End"), then the command

```
GEN_POLYGON name file1
```

creates the file "name.inp" of the desired polygonal grating.

If an input file for a grating determined by two polygonal profile lines is needed (cf. Figure 12), then this can be accomplished by calling the executable `GEN_POLYGON2` from the subdirectory `GEOMETRIES`. More precisely, if the file "`GEOMETRIES`/file1" contains the corner points of a polygonal profile line (in "`GEOMETRIES`/file1": in each line beginning without '#' there should be the x- and y-coordinate of one of the consecutive corner points, first point with x-coordinate 0, last point with x-coordinate 1, same y-coordinate for first and last point, all x-coordinates between 0 and 1, at least two different y-coordinates, last line should be "End") and if the file "`GEOMETRIES`/file2" contains the corner points of a second polygonal profile line (in "`GEOMETRIES`/file2": in each line beginning without '#' there should be the x- and y-coordinate of one of the consecutive corner points, first and last point must be corner of first polygon, second polygon must be on left-hand side of first, one to one correspondence of the corners on the two polygons between first and last point of second polygon[11], quadrilateral domain between corresponding segments on the left of first polygon, these quadrilaterals must be disjoint, last line should be "End"), then the command

```
GEN_POLYGON2 name file1 file2
```

creates the file "name.inp" of the desired polygonal grating.

If an input file for a grating determined by profile line given as $\{(f_x(t), f_y(t)) : 0 \leq t \leq 1\}$ is needed, then this can be accomplished by calling the executable `GEN_PROFILE` from the subdirectory `GEOMETRIES`. More precisely, suppose the profile line $\{(f_x(t), f_y(t)) : 0 \leq t \leq 1\}$ is given by the functions $t \mapsto f_x(t)$ and $t \mapsto f_y(t)$ defined by the c-code in the file "`GEOMETRIES`/profile.c". Then

```
GEN_PROFILE name 0.06
```

creates the file "name.inp" of the desired profile grating, where the profile curve is approximated by a polygonal line with a stepsize equal to 0.06 times the length of period (cf. Figure 13 where $f_x(t) = t$ and $f_y(t) = \{1.5 + 0.2\exp(\sin(6\pi t)) + 0.3\exp(\sin(8\pi t))\}/\{2\pi\}$).

If an input file for a grating determined by more than one non-intersecting and periodic profile lines given as $\{(f_x(j,t), f_y(j,t)) : \; 0 \leq t \leq 1\},\; j, = 1, \dots, n$ is needed, then this can be accomplished by calling the executable `GEN_PROFILES` from the subdirectory `GEOMETRIES`. More precisely, suppose n and the profile lines $\{(f_x(j,t), f_y(j,t)) : 0 \leq t \leq 1\}$ are given by the c-code in the file "`GEOMETRIES`/profiles.c". Then

```
GEN_PROFILES name 0.06
```

[11]To make it precise, suppose P and Q are the common corner points of the two polygonal curves and that R^1_1, R^1_2, ... , R^1_m are the consecutive corner points between P and Q on the first polygonal line and R^2_1, R^2_2, ... , R^2_n those on the second polygonal. Then the code requires $m = n$ and that the coating area between the two polygonal lines is the disjoint union of the triangle $PR^1_1R^2_1$, the quadrilaterals $R^1_1R^1_2R^2_2R^2_1$, ... , $R^1_{(m-1)}R^1_mR^2_mR^2_{(m-1)}$, and the triangle $R^1_mQR^2_m$.

creates the file "name.inp" of the desired profile grating, where the profile curves are approximated by a polygonal line with a stepsize equal to 0.06 times the length of period (cf. Figure 16 where $n = 3$, $f_x(j,t) = t$, $f_y(1,t) = \sin(2\pi t)$, $f_y(2,t) = \sin(2\pi t) - 0.5$, and $f_y(3,t) = \sin(2\pi t) - 1$).

If an input file for a pin grating determined by a simple non-intersecting profile line given as $\{(f_x(t), f_y(t)) : \ 0 \leq t \leq 1\}$ is needed[12], then this can be accomplished by calling the executable `GEN_PIN` from the subdirectory `GEOMETRIES`. More precisely, suppose x_{min} and the profile line $\{(f_x(t), f_y(t)) : 0 \leq t \leq 1\}$ are given by the c-code in the file "`GEOMETRIES`/pin.c". Then

```
GEN_PIN name 0.06
```

creates the file "name.inp" of the desired profile grating, where the profile curves are approximated by a polygonal line with a stepsize equal to 0.06 times the length of period (cf. Figure 17 where $x_{min} = 0.2$, $f_x(t) = x_{min} + (1 - 2x_{min})\,t$, and $f_y(t) = 0.5\,\sin(\pi t)$).

If an input file for a coated pin grating determined by the simple non-intersecting profile lines given as $\{(f_x(j,t), f_y(j,t)) : 0 \leq t \leq 1\}$, $j = 1,2$ is needed[13], then this can be accomplished by calling the executable `GEN_CPIN` from the subdirectory `GEOMETRIES`. More precisely, suppose x_{min}, a_1, a_2 and the profile lines $\{(f_x(j,t), f_y(j,t)) : 0 \leq t \leq 1\}$ j=1,2 are given by the c-code in the file "`GEOMETRIES`/cpin.c". Then

```
GEN_CPIN name 0.06
```

creates the file "name.inp" of the desired profile grating, where the profile curves are approximated by a polygonal line with a stepsize equal to 0.06 times the length of period (cf. Figure 18 where $x_{min} = 0.2$, $a_1 = 0.2$, $a_1 = 0.8$, $f_x(1,t) = x_{min} + (1 - 2x_{min})\,t$, $f_y(1,t) = 0.5\,\sin(\pi t)$, and $\{(f_x(2,t), f_y(2,t)) : \ 0 \leq t \leq 1\}$ is the polygonal curve connecting the four points $(f_x(1,a_1), f_y(1,a_1))$, $(f_x(1,a_1), f_y(1,0.5){+}0.1)$, $(f_x(1,a_2), f_y(1,0.5){+}0.1)$, and $(f_x(1,a_2), f_y(1,a_2)))$.

If an input file for a coated pin grating of type 2 determined by the simple non-intersecting profile lines given as $\{(f_x(j,t), f_y(j,t)) : 0 \leq t \leq 1\}$, $j = 1,2$ is needed[14],

[12]I.e., over a flat grating with surface $\{(x,0) : \ 0 \leq x \leq 1\}$ a material part is attached which is located between $\{(x,0) : 0 \leq x \leq 1\}$ and $\{(f_x(t), f_y(t)) : 0 \leq t \leq 1\}$. Here $\{(f_x(t), f_y(t)) : 0 \leq t \leq 1\}$ is a simple open arc connecting $(f_x(0), f_y(0)) = (x_{min}, 0)$ with $(f_x(1), f_y(1)) = (1 - x_{min}, 0)$ such that $0 < x_{min} < 0.5$ is a fixed number, such that $0 < f_x(t) < 1$, $0 < t < 1$, and such that $0 < f_y(t)$, $0 < t < 1$.

[13]I.e., over a flat grating with surface $\{(x,0) : 0 \leq x \leq 1\}$ a material part is attached which is located between $\{(x,0) : 0 \leq x \leq 1\}$ and $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$. Here $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ is a simple open arc connecting $(f_x(1,0), f_y(1,0)) = (x_{min}, 0)$ with $(f_x(1,1), f_y(1,1)) = (1 - x_{min}, 0)$ such that $0 < x_{min} < 0.5$ is a fixed number, such that $0 < f_x(1,t) < 1$, $0 < t < 1$, and such that $0 < f_y(1,t)$, $0 < t < 1$. Additionally, a coating layer is attached located between the first curve $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ and a second curve $\{(f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$. The last connects the point $(f_x(1,a_1), f_y(1,a_1)) = (f_x(2,0), f_y(2,0))$ with $(f_x(1,a_2), f_y(1,a_2)) = (f_x(2,1), f_y(2,1))$. Moreover, $\{(f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$ is a simple open arc above $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ such that $0 < f_x(2,t) < 1$, $0 < t < 1$.

[14]I.e., over a flat grating with surface $\{(x,0) : \ 0 \leq x \leq 1\}$ a material part is attached which is located between the line $\{(x,0) : \ 0 \leq x \leq 1\}$ and $\{(f_x(1,t), f_y(1,t)) : \ 0 \leq t \leq 1\}$. Here $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ is a simple open arc connecting $(f_x(1,0), f_y(1,0)) = (x_{min}, 0)$ with $(f_x(1,1), f_y(1,1)) = (1 - x_{min}, 0)$ such that $0 < x_{min} < 0.5$ is a fixed number, such that $0 < f_x(1,t) < 1$, $0 < t < 1$, and such that $0 < f_y(1,t)$, $0 < t < 1$. Additionally, a coating layer is attached located between the first curve $\{(f_x(1,t), f_y(1,t)) : \ 0 \leq t \leq 1\}$ united with the the two straight line segments $\{(x,0) : \ x_1 \leq x \leq x_{min}\}$ and $\{(x,0) : \ 1 - x_{min} \leq x \leq x_2\}$ and a second curve $\{(f_x(2,t), f_y(2,t)) : \ 0 \leq t \leq 1\}$. The last connects

then this can be accomplished by calling the executable `GEN_CPIN2` from the subdirectory `GEOMETRIES`. More precisely, suppose x_{min} and the profile lines $\{(f_x(j,t), f_y(j,t)) : 0 \leq t \leq 1\}$ j=1,2 are given by the c-code in the file "`GEOMETRIES`/cpin2.c". Then

```
GEN_CPIN2 name 0.05
```

creates the file "name.inp" of the desired profile grating, where the profile curves are approximated by a polygonal line with a stepsize equal to 0.05 times the length of period (cf. Figure 19 where $x_{min} = 0.2$, $f_x(1,t) = x_{min} + (1 - 2x_{min})\,t$, $f_y(1,t) = 0.5 \sin(\pi t)$, and $f_x(2,t) = 0.5\,x_{min} + (1 - x_{min})\,t$, $f_y(2,t) = 0.8 \sin(\pi t)$).

In some applications the user might think that meshes graded towards the corner points of the polygonal interfaces should improve the convergence behaviour of the FEM solution. Such graded meshes can be generated by adding clouds of points to the neighbourhood of the corners. If a geometry is given by the input file "name.inp", then an advanced input file "name+.inp" including point clouds at the corners can be generated by the command

```
POINT_CLOUD_INP name 30 5 100.
```

Here the last argument 100 is the angle in degrees such that "name+.inp" contains point clouds at all interface corners of "name.inp" with angles less than 100°. The third argument 5 is the number of layers in the point clouds, i.e. the cloud points are located at circular curves around the corner with radii $\varrho * 0.5$, $\varrho * 0.5^2$, $\ldots$, $\varrho * 0.5^5$. The positive real ϱ is chosen as large as possible such that the point clouds do not intersect other point clouds or corner points. Finally, the second argument 30 is the number of points at each circular curve. In other words, the cloud points are the intersections of the circular curves with rays through the corner points such that the angle between two neighbour rays is about 360°/30.

3.3 Graded `FEM`-mesh generated through geometry input file

The geometry is read from the input file "name1.inp" and the mesh generator will create a finite element partition with given refinement level. The generator tries to compute regular (almost uniform) triangles. Over the generated finite element mesh the trial functions and approximate solutions are determined.

Sometimes the field solution of the problem is difficult to approximate by the trial functions defined over regular meshes. For instance, the field solutions may have singular points at corners and surface layers close to the interfaces. If the corresponding field components by the executables `GFEM_PLOT` resp. `FEM_PLOT` are visualised and if the the isoline mode is switched on (cf. Subsection 5.3), then a huge number of isoline curves surrounding a point indicate a singularity point. Many isolines located close and parallel to the interface lines indicate surface layers. For these cases, graded meshes can enhance the approximation of the `FEM` and `GFEM`.

A mesh grading towards a corner point can be enforced if additional points close to

the first point $(x_1, 0) = (f_x(2,0), f_y(2,0))$ with the last point $(x_2, 0) = (f_x(2,1), f_y(2,1))$. Moreover, $\{(f_x(2,t), f_y(2,t)) : \ 0 \leq t \leq 1\}$ is a simple open arc above $\{(f_x(1,t), f_y(1,t)) : \ 0 \leq t \leq 1\}$ such that $0 < f_x(2,t) < 1$, $0 < t < 1$. The functions $f_x(1,.)$, $f_x(2,.)$, $f_y(1,.)$, and $f_y(2,.)$ and the parameter x_{min} are defined by the code of the file "../`GEOMETRIES`/cpin2.c".

the corners are introduced in the input file "name1.inp". We recommend to approach the corner by these additional points from one, two or three directions at distances of size $0.1 * 2^l,\ l = 1, 2, \ldots$. These points need not to be included into the list of triangle corners. The mesh generator, however, will include the points into the finite element mesh such that a grading towards the corner is achieved. Note that mesh gradings toward points does not increase the overall number of mesh points, essentially.

A mesh grading towards an interface line can be enforced if a tiny additional layer on one side of the interface is introduced by the geometry input file "name1.inp". This layer must have a different index of material. Of course, the refractive index chosen for this material index in the data file "name.dat" must be the same as for the material from which the layer is artificially separated. The mesh generator resolves the tiny layer structure by small regular triangles and the size of the triangles close to the layer grows slowly with the distance to the layer. Unfortunately, the overall number of mesh points notably increases if the width of the additional layer decreases. Nevertheless, the approximation using a graded mesh is often better than that of a comparable refined uniform mesh.

Finally, we should mention the different level refinement strategies for regular and graded meshes. The standard way of mesh refinement is to increase the level in the input file "name.dat" (cf. Subsection 5.2). If the level l is large such that the maximal meshsize $h_0\, 2^{1-l}$ is less than the minimal distance of the additional points to the singular point and less than the width of the additional layer, then the generated mesh is regular again. In other words, the degree of grading of the meshes is reduced by increasing the level of discretization. Alternatively to increasing the level by one, the parameter n_{UPA} of the control files "generalised.Dat" and "conical.Dat" (cf. Sections 6 and 7) can be doubled. In this case the degree of grading of the meshes is maintained.

3.4 Input file "name1.inp" by TGUI

The easiest way to create a geometry input file "name1.inp" is to use the graphical user interface program `TGUI` in the subdirectory `GEOMETRIES`. Just call `TGUI` and draw one period of the cross section of the grating. Note that `TGUI` is equipped with a complete help system. For the special `DIPOG-2.1` format, we mention the following.

The dimensions of the cross section details including the period (horizontal dimension) of the grating are measured in nano meters. Typically, the vertical and horizontal diameters are about 1 000 nano meters. Note, however, that the geometry will be scaled to period 1 for the input format and the actual period is fixed in the data file "name.dat" (cf. Sects. 3.1 and 5.1).

The cross section domain is bounded by two lateral sides as well as an upper and lower boundary curve connecting the upper resp. lower points of the lateral sides. Possibly, this domain is split into several material parts by polygonal interfaces. If the upper or the lower curve is a horizontal straight line segment, then `DIPOG` can add additional upper and lower rectangular strips to the geometry. However, these strips are described in the data file "name.dat" (cf. Sect. 5.1) and not by the geometry input format "name1.inp".

The lateral sides of the cross section domain must form straight line segments in exactly vertical direction. The segment indicator of the segments of these sides must be 3, whereas the segments forming the upper and lower boundaries get indicator 2 and 1, respectively.

The indices of the material parts can be fixed to any positive integer. Suppose that n_{mat} is the maximal index. Then the materials will be determined in a list of n_{mat} refractive indices (optical indices) given in the data file "name.dat" of DIPOG (cf. Sect. 5.1). Note that index 1 is reserved for the refractive index of the material adjacent to the cross section domain from the upper side (either cover material or lowest upper additional strip) and index n_{mat} for the refractive index of the material adjacent to the cross section domain from the lower side (either substrate material or highest lower additional strip).

The domain for the FEM computation is a rectangle which is a slight extension of the just mentioned cross section domain into the upper and lower direction. The additional strips are possibly not contained in the FEM domain since they are treated via boundary operators. Even if the cross section domain is rectangular, small strips are added in order to enable a fast treatment of the boundary operators.

However, in case of classical TE polarization and if the upper resp.lower boundary curves are horizontal straight lines of the cross section domain, the extension to the FEM domain by small upper resp. lower strips can be suppressed (cf. Sect. 3.2). To this end the indicators of the boundary segments must be changed from 2 resp. 1 to 5 resp. 4. Moreover, since the fast treatment of the boundary operators requires uniform partitions, the node points on the upper resp. lower boundary lines must be rational. More precisely, for left and right end points A and B, the node points must be of the form $A + (B - A) * r$ with $r = p/q$ and $0 < p < q < 1\,000$.

3.5 Code words to indicate special geometries

One can indicate special grating geometries in the input file "name.dat" by special code words. We explain these here.

```
# Grating data:
  echellea R 0.3 0.03 0.04
```

indicates an ECHELLE GRATING TYPE A (right-angled triangle with hypotenuse parallel to the direction of the periodicity, right interior angle greater than 45°, cf. Figure 6) with depth of 0.3 μm (i.e., triangle height = 0.3 μm) and with coated layers of height 0.03 μm resp. 0.04 μm over the first resp. second part of the grating (measured in direction perpendicular to the echelle profile, height greater or equal to zero).

```
# Grating data:
  echellea L 0.3 0.03 0.04
```

indicates an ECHELLE GRATING TYPE A (right-angled triangle with hypotenuse parallel to the direction of the periodicity, left interior angle greater than 45°) with parameters like above.

```
# Grating data:
  echellea A 60. 0.03 0.04
```

indicates an ECHELLE GRATING TYPE A (right-angled triangle with hypotenuse parallel to the direction of the periodicity) with left interior angle $\alpha = 60°$ (i.e. the depth is equal to the period multiplied by $\sin(\alpha)\cos(\alpha)$) and other parameters like above.

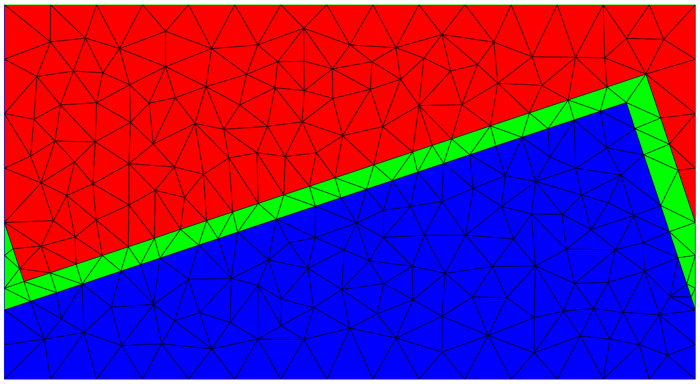


Figure 6: Echelle grating of type A.

```
# Grating data:
  echelleb 60. 0.05
```

indicates an Echelle Grating Type B (right-angled triangle with one of the legs parallel to the direction of the periodicity, cf. Figure 7) with angle 60° (angle enclosed by hypotenuse and by the leg parallel to the period) and with a coated layer of height 0.05 μm (measured in direction perpendicular to echelle profile, height greater or equal to zero).

```
# Grating data:
  echelle L 100. R 30. 0.05 0.1
```

indicates a General Echelle Grating with a left blaze angle of 100°, with a right blaze angle of 30°, with a coated layer over the left blaze side of height 0.05 μm (measured in direction perpendicular to the echelle profile, height greater or equal to zero) and with a coated layer over the right blaze side of height 0.1 μm (measured in direction perpendicular to the echelle profile, height greater or equal to zero, must be zero if the previous height is zero). Instead of the two inputs “L 100.” and “R 30.” one can choose also the inputs “A 90.” for an apex angle of 90° or “D 0.2” for a depth of the grating equal to 0.2 μm. Moreover, any combination of two inputs of the types “A 90.”, “L 110.”, “R 90.”, and “D 0.2” is accepted. However, the choice “A 90.” and “D 0.2” might be ambiguous. By definition it fixes an echelle grating with right blaze angle larger than the left. To get the flipped grating with left blaze angle larger than the right, the input should be “A 90.” and “D -0.2”. Figure 6 corresponds to a period of 1 μm and to the code words “echelle A 90. D 0.3 0.03 0.04”.

```
# Grating data:
  trapezoid 60. 0.6 3 0.2 0.1 0.1 0.05
```

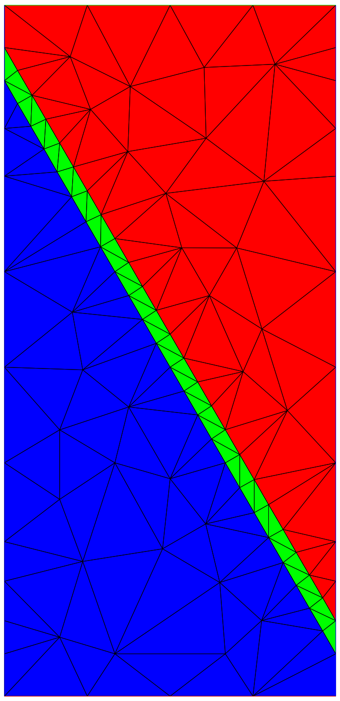


Figure 7: Echelle grating of type B.

indicates a Trapezoidal Grating (isosceles trapezoid with the basis parallel to the direction of the periodicity, cf. Figure 8) with angle of 60° (angle enclosed by basis and the sides) with a basis of length 0.6 μm consisting of 3 material layers of heights 0.2 μm, 0.1 μm, and 0.1 μm, respectively, and with a coated layer of height 0.05 μm (greater or equal to zero)

```
# Grating data:
  mtrapezoid
  2
  0.02 0.06
  90. 80.
  0.04
  3
  0.16 0.21 0.18
  0.25 0.50 0.75
```

indicates a Multi Trapezoidal Grating (trapezoids with bases parallel to the direction of the periodicity, cf. Figure 9) each trapezoid consists of 2 layers with height 0.02 μm and 0.06 μm, respectively; side-wall angle of these trapezoidal layers are 90° and 80°; the number of trapezoids is 3 and the lateral width of the trapezoids measured at a height of 0.04 μm is 0.16 μm, 0.21 μm, and 0.18 μm, respectively; finally the midpoints of the trapezoids have the lateral distances 0.25 μm, 0.50 μm, and 0.75 μm to the starting point of a period.

```
# Grating data:
  lAmellar 3 4
  0.2 0.6
  -0.2 1.0
```

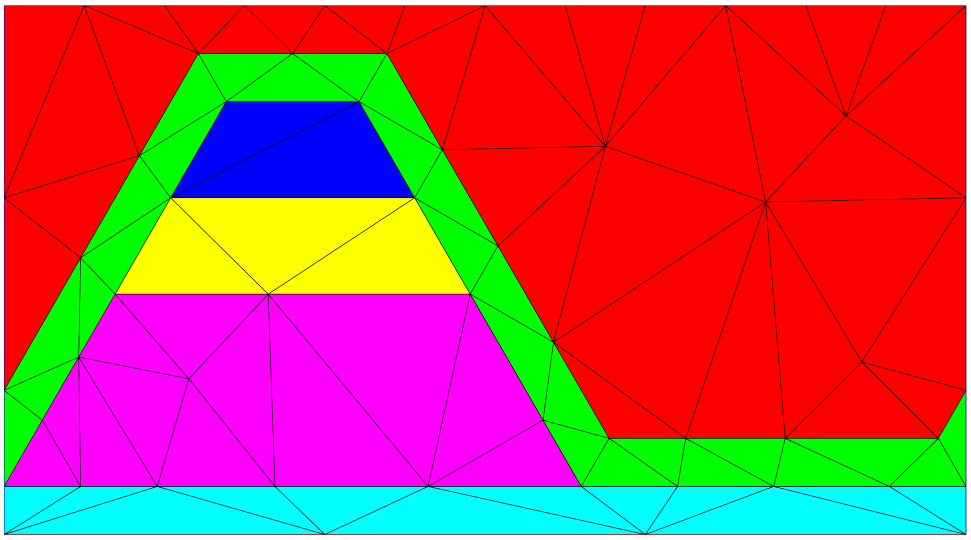


Figure 8: Trapezoidal grating.

```
0. 0.5 0.7
0.0 0.50 0.90
0.00 0.500 0.900
```

indicates a LAMELLAR GRATING (rectangular grating consisting of several materials placed in rectangular subdomains, cf. Figure 10) with 3 columns each divided into 4 rectangular layers, first column with x-coordinate in 0 μm<x<0.2 μm, second column with 0.2 μm<x<0.6 μm, third column with 0.6 μm<x<period (period given above), whole grating with y-coordinate s.t. -0.2 μm<y<1.0 μm, first column: first layer with -0.2 μm<y<0. μm, second with 0. μm<y<0.5 μm, third with 0.5 μm<y<0.7 μm and fourth with 0.7 μm<y<1. μm, second column: first layer with -0.2 μm<y<0.0 μm, second with 0.0 μm<y<0.50 μm, third with 0.50 μm<y<0.90 μm and fourth with 0.90 μm<y<1. μm, third column: first layer with -0.2 μm<y<0.00 μm, second with 0.00 μm<y<0.500 μm, third with 0.500 μm<y<0.900 μm and fourth with 0.900 μm<y<1. μm .

```
# Grating data:
   lAmellar 1 1
   0.2 0.8
```

indicates a SIMPLE LAYER (special case of lamellar grating) with layer material s.t. y-coordinate satisfies 0.2 μm<y<0.8 μm.

```
# Grating data:
```

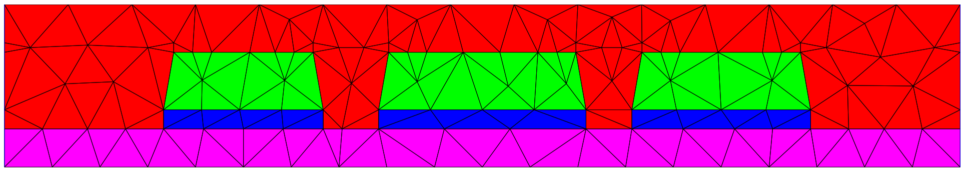


Figure 9: Multi-trapezoidal grating.

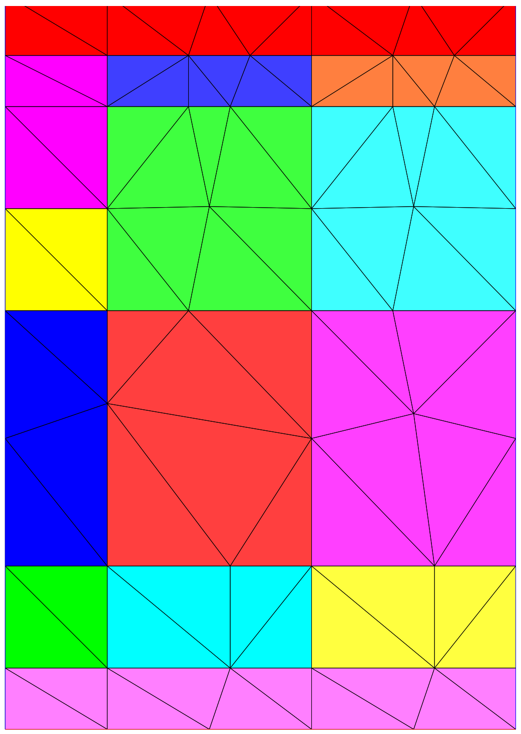


Figure 10: Lamellar grating.

polygon file1

indicates a Grating Determined by a Polygonal Line (cf. Figure 11) defined by the data in the file with name "`GEOMETRIES`/file1". The x- and y-coordinates of the points in "`GEOMETRIES`/file1" are supposed to be scaled such that the period is one (in "`GEOMETRIES`/file1": in each line beginning without '#' there should be the x- and y-coordinate of one of the consecutive corner points, first point with x-coordinate 0, last point with x-coordinate 1, same y-coordinate for first and last point, all x-coordinates between 0 and 1, at least two different y-coordinates, last line should be "End"). If the program does not find the file "`GEOMETRIES`/file1", then it takes the file "file1" of the current working directory.

```
# Grating data:
  polygon2 file1 file2
```

indicates a Coated Grating Determined by Polygonal Lines (cf. Figure 12), i.e. the grating profile line is defined by the data in the file with name "`GEOMETRIES`/file1" (in "`GEOMETRIES`/file1": in each line beginning without '#' there should be the x- and y-coordinate of one of the consecutive corner points, first point with x-coordinate 0, last point with x-coordinate 1, same y-coordinate for first and last point, all x-coordinates between 0 and 1, at least two different y-coordinates, last line should be "End") and the coated layer is enclosed between the polygonal line of "`GEOMETRIES`/file1" and the polygonal line of the file with name "`GEOMETRIES`/file2" (in "`GEOMETRIES`/file2": in each line beginning without '#' there should be the x- and y-coordinate of one of the consecutive corner points, first and last point must be corner of first polygon, second polygon must be on left-hand

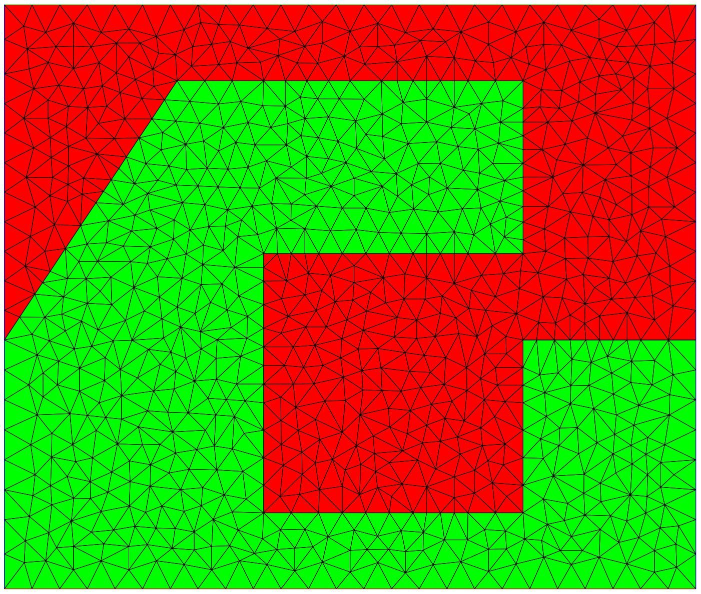


Figure 11: Grating determined by a polygonal line.

side of first, one to one correspondence of the corners on the two polygons between first and last point of second polygon[15], quadrilateral domain between corresponding segments on the left of first polygon, these quadrilaterals must be disjoint, last line should be "End"). The x- and y-coordinates of the points in "`GEOMETRIES`/file1" and "`GEOMETRIES`/file2" are supposed to be scaled such that the period is one. If the program does not find the files "`GEOMETRIES`/file1" and "`GEOMETRIES`/file2", then it takes the files "file1" and "file2" of the current working directory.

```
# Grating data:
  profile
```

indicates a Grating Determined by a Smooth Parametric Curve, i.e. grating determined by profile line given as $\{\text{period} \cdot (f_x(t), f_y(t)) : \ 0 \le t \le 1\}$, where the functions $t \mapsto f_x(t)$ and $t \mapsto f_y(t)$ are defined by the c-code of the file "`GEOMETRIES`/profile.c" (cf. Figure 13 where $f_x(t) = t$ and $f_y(t) = \{1.5 + 0.2 \exp(\sin(6\pi t)) + 0.3 \exp(\sin(8\pi t))\}/\{2\pi\}$). If the program does not find the file "`GEOMETRIES`/profile.c", then it takes the file "profile.c" of the current working directory.

```
# Grating data:
  profile_par 2 3
```

[15]To make it precise, suppose P and Q are the common corner points of the two polygonal curves and that R^1_1, R^1_2, ... , R^1_m are the consecutive corner points between P and Q on the first polygonal line and R^2_1, R^2_2, ... , R^2_n those on the second polygonal. Then the code requires $m = n$ and that the coating area between the two polygonal lines is the disjoint union of the triangle $PR^1_1R^2_1$, the quadrilaterals $R^1_1R^1_2R^2_2R^2_1$, ... , $R^1_{(m-1)}R^1_mR^2_mR^2_{(m-1)}$, and the triangle $R^1_mQR^2_m$.

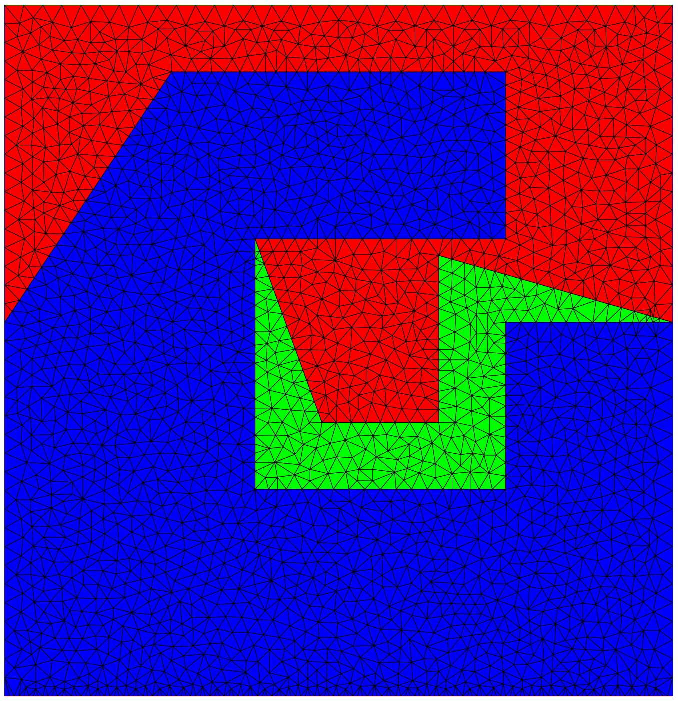


Figure 12: Grating determined by polygonal lines.

```
1
0
1.5
0.2
0.3
```

indicates a Grating Determined by a Smooth Parametric Curve, With Parameters, i.e. grating determined by profile line given as $\{\text{period} \cdot (f_x(t), f_y(t)) : \; 0 \leq t \leq 1\}$, where the functions $t \mapsto f_x(t)$ and $t \mapsto f_y(t)$ are defined by the c-code of the file "`GEOMETRIES`/profile_par.c". The last code uses 2 integer parameters and 3 real parameters named IPARaM1, IPARaM2, RPARaM1, RPARaM2, and RPARaM3. The integer parameters take the values 1 and 0 following the first line of the calling sequence and the real parameters take the values 0.15, 1., and 0. following the integer parameter values (cf. Figure 13 where $f_x(t) = t$ and $f_y(t) = \{1.5 + 0.2 \exp(\sin(6\pi t)) + 0.3 \exp(\sin(8\pi t))\}/\{2\pi\}$, parameter 1 is the index of the curve chosen from "`GEOMETRIES`/profile_par.c", parameter 0 is the number of corners of the curve, parameters 1.5, 0.2, and 0.3 are scaling parameters in the y-coordinate of the curve). Note that any number of parameters is possible for a corresponding file "`GEOMETRIES`/profile_par.c". If the program does not find the file "`GEOMETRIES`/profile_par.c", then it takes the file "profile_par.c" of the current working directory.

```
# Grating data:
    profile 0.125 * sin(2. * M_PI * t)
```

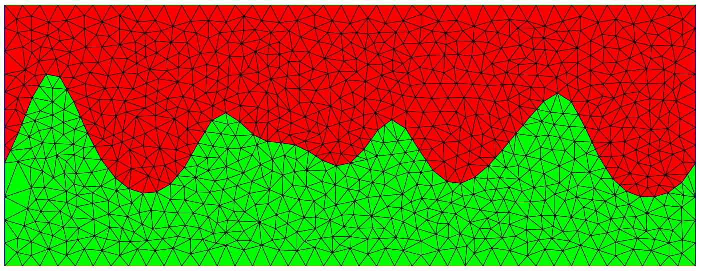


Figure 13: Grating determined by a smooth parametric curve.

indicates a GRATING DETERMINED BY A SIMPLE SMOOTH FUNCTION (cf. Figure 14), i.e. grating determined by a profile line given as $\{\text{period} \cdot (t, f_y(t)) : \ 0 \le t \le 1\}$, where the function $t \mapsto f_y(t)$ is defined by the c-code $f_y(t) = 0.125 \sin(2\pi t)$, (do not use any blank/space in the c-code).

# Grating data:
profile $0.5 + 0.5 * \cos(\text{M\_PI} * (1. - t))$ $0.25 * \sin(\text{M\_PI} * t)$

indicates a GRATING DETERMINED BY A SIMPLE SMOOTH PARAMETRIC CURVE (cf. Figure 15), i.e., grating determined by ellipsoidal profile line given as $\{\text{period} \cdot (f_x(t), f_y(t)) : 0 \le t \le 1\}$, where the functions $t \mapsto f_x(t)$ and $t \mapsto f_y(t)$ are defined by the c-codes $f_x(t) = 0.5 + 0.5 \cos(\pi(1 - t))$ and $f_y(t) = 0.25 \sin(\pi t)$, respectively (no blank/space in c-code!).

# Grating data:
profiles

indicates a GRATING DETERMINED BY SMOOTH PARAMETRIC CURVES, i.e. grating determined by n non-intersecting and periodic profile lines given from above to below as $\{\text{period} \cdot (f_x(j,t), f_y(j,t)) : \ 0 \le t \le 1\}$, $j = 1, \ldots, n$, where n and the functions $t \mapsto f_x(j,t)$ and $t \mapsto f_y(j,t)$ are defined by the c-code of the file "`GEOMETRIES`/profiles.c" (cf. Figure 16 where $n = 3$, $f_x(j,t) = t$, $f_y(1,t) = \sin(2\pi t)$, $f_y(2,t) = \sin(2\pi t) - 0.5$, and $f_y(3,t) = \sin(2\pi t) - 1$). If the program does not find the file "`GEOMETRIES`/profiles.c", then it takes the file "profiles.c" of the current working directory.

# Grating data:

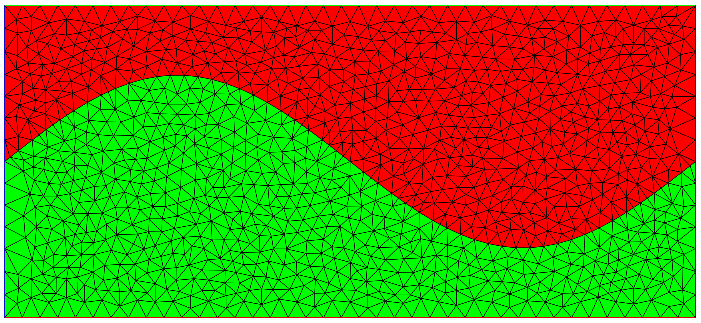


Figure 14: Grating determined by a simple smooth function.

```
profiles_par 1 2
3
0.5
0.50
```

indicates a Grating Determined by Smooth Parametric Curves, With Parameters, i.e. grating determined by n non-intersecting and periodic profile lines given from above to below as $\{\text{period} \cdot (f_x(j,t), f_y(j,t)) : \ 0 \le t \le 1\}$, $j = 1, \ldots, n$, where n and the functions $t \mapsto f_x(j,t)$ and $t \mapsto f_y(j,t)$ are defined by the c-code included in the file "GEOMETRIES/profiles_par.c". The last code uses 1 integer parameter and 2 real parameters named IPARaM1, RPARaM1, and RPARaM2. The integer parameter takes the value 3 following the first line of the calling sequence and the real parameters take the values 0.5 and 1. following the integer parameter values (cf. Figure 16 where $n = 3$, $f_x(j,t) = t$, $f_y(1,t) = \sin(2\pi t)$, $f_y(2,t) = \sin(2\pi t) - 0.5$, and $f_y(3,t) = \sin(2\pi t) - (0.5 + 0.50)$, parameter 3 is the number n of boundary and interface curves, parameter 0.5 is the width of the first layer and parameter 0.50 that of the second). Note that any number of parameters is possible for a corresponding file "GEOMETRIES/profiles_par.c". If the program does not find the file "GEOMETRIES/profiles_par.c", then it takes the file "profiles_par.c" of the current working directory.

```
# Grating data:
  pin
```

indicates a Pin Grating Determined By Parametric Curve (cf. Figure 17 where $x_{min} = 0.2$, $f_x(t) = x_{min} + (1 - 2x_{min})\, t$, and $f_y(t) = 0.5\, \sin(\pi t)$), i.e. over a flat grating

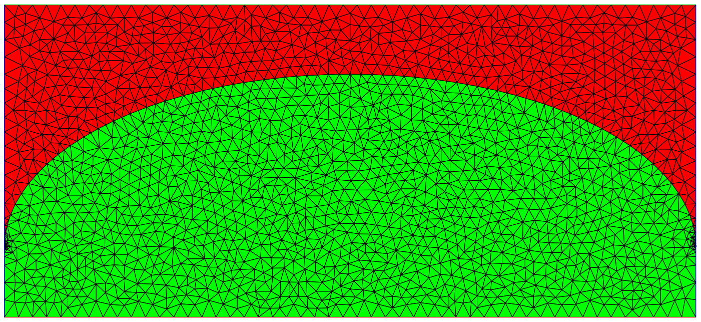


Figure 15: Grating determined by a simple smooth parametric curve.

with surface $\{\text{period} \cdot (x,0) : 0 \leq x \leq 1\}$ a material part is attached which is located between $\{\text{period} \cdot (x,0) : 0 \leq x \leq 1\}$ and $\{\text{period} \cdot (f_x(t), f_y(t)) : 0 \leq t \leq 1\}$. Here $\{(f_x(t), f_y(t)) : 0 \leq t \leq 1\}$ is a simple open arc connecting $(f_x(0), f_y(0)) = (x_{min}, 0)$ with $(f_x(1), f_y(1)) = (1 - x_{min}, 0)$ such that $0 < x_{min} < 0.5$ is a fixed number, such that $0 < f_x(t) < 1$, $0 < t < 1$, and such that $0 < f_y(t)$, $0 < t < 1$. The functions f_x, f_y and the parameter x_{min} are defined by the code in "../`GEOMETRIES`/pin.c". If the program does not find the file "`GEOMETRIES`/pin.c", then it takes the file "pin.c" of the current working directory.

```
# Grating data:
  cpin
```

indicates a Coated Pin Grating Determined By Two Parametric Curves (cf. Figure 18 where $x_{min} = 0.2$, $a_1 = 0.2$, $a_1 = 0.8$, $f_x(1,t) = x_{min} + (1 - 2x_{min})\,t$, $f_y(1,t) = 0.5\,\sin(\pi t)$, and $\{(f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$ is the polygonal curve connecting the four points $(f_x(1,a_1), f_y(1,a_1))$, $(f_x(1,a_1), f_y(1,0.5) + 0.1)$, $(f_x(1,a_2), f_y(1,0.5) + 0.1)$, and $(f_x(1,a_2), f_y(1,a_2)))$, i.e. over a flat grating with surface $\{\text{period} \cdot (x,0) : 0 \leq x \leq 1\}$ a material part is attached which is located between $\{\text{period} \cdot (x,0) : 0 \leq x \leq 1\}$ and $\{\text{period} \cdot (f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$. Here $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ is a simple open arc connecting $(f_x(1,0), f_y(1,0)) = (x_{min}, 0)$ with $(f_x(1,1), f_y(1,1)) = (1 - x_{min}, 0)$ such that $0 < x_{min} < 0.5$ is a fixed number, such that $0 < f_x(1,t) < 1$, $0 < t < 1$, and such that $0 < f_y(1,t)$, $0 < t < 1$. Additionally, a coating layer is attached located between the first curve $\{\text{period} \cdot (f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ and a second curve $\{\text{period} \cdot (f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$. The last connects the

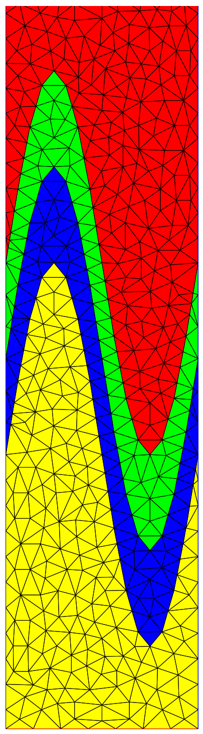


Figure 16: Grating determined by smooth parametric curves.

first point period $\cdot$ $(f_x(1,a_1), f_y(1,a_1))$ = period $\cdot$ $(f_x(2,0), f_y(2,0))$ with the last point period$\cdot$$(f_x(1,a_2), f_y(1,a_2))$ = period$\cdot$$(f_x(2,1), f_y(2,1))$. Moreover, $\{(f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$ is a simple open arc above $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ such that $0 < f_x(2,t) < 1$, $0 < t < 1$. The functions $f_x(1,.)$, $f_x(2,.)$, $f_y(1,.)$, and $f_y(2,.)$ and the parameters a_1, a_2, and x_{min} are defined by the code of the file "../`GEOMETRIES`/cpin.c". If the program does not find the file "`GEOMETRIES`/cpin.c", then it takes the file "cpin.c" of the current working directory.

```
# Grating data:
  cpin2
```

indicates a Coated Pin Grating Determined By Two Parametric Curves, Type 2 (cf. Figure 19 where $x_{min} = 0.2$, $f_x(1,t) = x_{min} + (1 - 2x_{min})\,t$, $f_y(1,t) = 0.5\,\sin(\pi t)$, and $f_x(2,t) = 0.5\,x_{min} + (1 - x_{min})\,t$, $f_y(2,t) = 0.8\,\sin(\pi t)$), i.e. over a flat grating with surface $\{\text{period} \cdot (x,0) : 0 \leq x \leq 1\}$ a material part is attached which is located between the line $\{\text{period} \cdot (x,0) : 0 \leq x \leq 1\}$ and $\{\text{period} \cdot (f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$. Here $\{(f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ is a simple open arc connecting $(f_x(1,0), f_y(1,0)) = (x_{min}, 0)$ with $(f_x(1,1), f_y(1,1)) = (1 - x_{min}, 0)$ such that $0 < x_{min} < 0.5$ is a fixed number, such that $0 < f_x(1,t) < 1$, $0 < t < 1$, and such that $0 < f_y(1,t)$, $0 < t < 1$. Additionally, a coating layer is attached located between the first curve $\{\text{period} \cdot (f_x(1,t), f_y(1,t)) : 0 \leq t \leq 1\}$ united with the the two straight line segments $\{\text{period} \cdot (x,0) : x_1 \leq x \leq x_{min}\}$ and $\{\text{period} \cdot (x,0) : 1 - x_{min} \leq x \leq x_2\}$ and a second curve $\{\text{period} \cdot (f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$. The last connects the first point period$\cdot$$(x_1,0)$ = period$\cdot$$(f_x(2,0), f_y(2,0))$ with the last point period$\cdot$$(x_2,0)$ = period$\cdot$$(f_x(2,1), f_y(2,1))$. Moreover, $\{(f_x(2,t), f_y(2,t)) : 0 \leq t \leq 1\}$

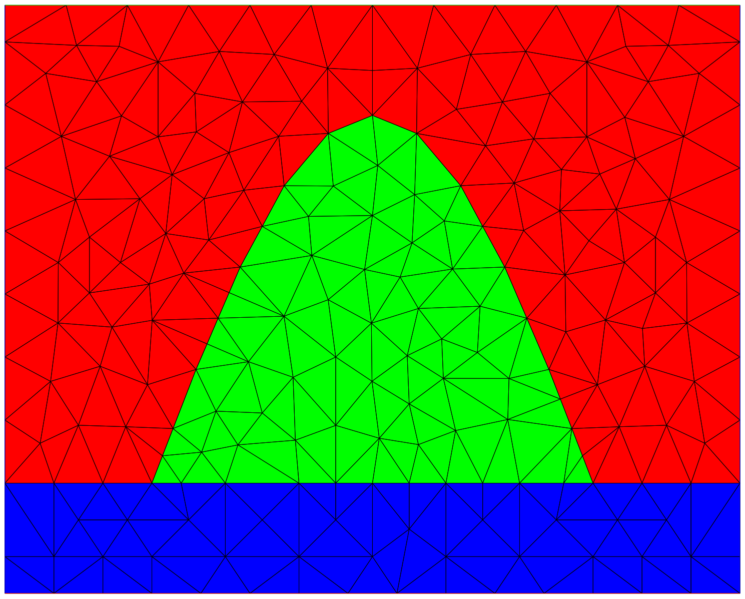


Figure 17: Pin grating determined by parametric curve.

is a simple open arc above $\{(f_x(1,t), f_y(1,t)) : \ 0 \le t \le 1\}$ such that $0 < f_x(2,t) < 1$, $0 < t < 1$. The functions $f_x(1,.)$, $f_x(2,.)$, $f_y(1,.)$, and $f_y(2,.)$ and the parameter x_{min} are defined by the code of the file "../GEOMETRIES/cpin2.c". If the program does not find the file "GEOMETRIES/cpin2.c", then it takes the file "cpin2.c" of the current working directory.

```
# Period of grating in micro meter:
   2.
# Grating data:
   bOX 0.1
   -1. 1.
   2 3
   { 2.*t }
   { -0.6 }
   { 1.+0.4*cos(2.*M_PI*t)}
   { 0.4*sin(2.*M_PI*t) }
   1. 0.8
   1. 0.
   1.-0.8
```

indicates a Box Grating (cf. Figure 20) where the box is $[0., period = 2.] \times [-1., 1.]$ given in μm. The number 0.1 behind the code word BOX is a factor for the mesh size of the FEM discretization of the box. The box is divided by 2 curves into 3 different material parts. The curves are given by the c-code in the following 2 times two lines. The first code line is the x-component of the first curve, the second the y-component of the first curve, the third

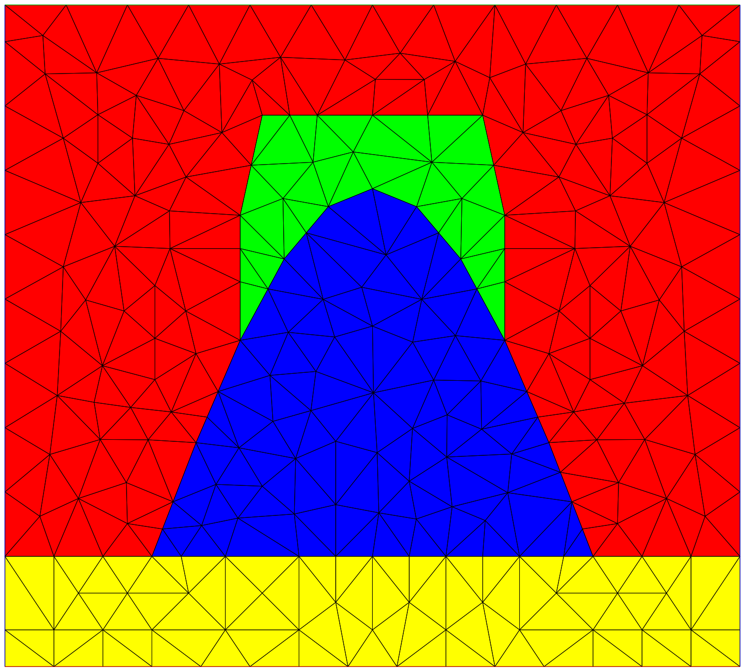


Figure 18: Coated pin grating determined by two parametric curves.

the x-component of the second curve, etc.. The last three input lines define material points. Material with index one is located in that part of the box which is separated by the above curves and contains the point (1.,0.8). Material with index two is located in that part of the box which is separated by the above curves and contains the point (1.,0.), etc.. Note that the curves must be simple not self intersecting, and they must be contained in the box. The only allowed intersection points of two different curves are the end points of the curves. Finally, the material areas separated by the curves in the box must be such that the area with the first index contains a whole strip under the upper boundary side of the box and that the area with the last index contains a whole strip over the lower boundary side. If these conditions are violated, then the computational result will be complete nonsense.

# Grating data:
rough_mls name

indicates a Multilayer System With Rough Interfaces (cf. (21)), i.e., n_{la} layers followed by n_{mld} times n_{ld} layers and followed by n_{lb} lowest layers. This system is described by the file "name", which is contained in the directory "../GEOMETRIES" (alternatively, "name" must contain the path of the file). This input file "name" contains the following ordered data each in a separate line (comment lines begin with '#'):

– dummy, file name
– width of additional layer above and below the structure (must be positive or could be 'no' for no additional layer)
– period of grating

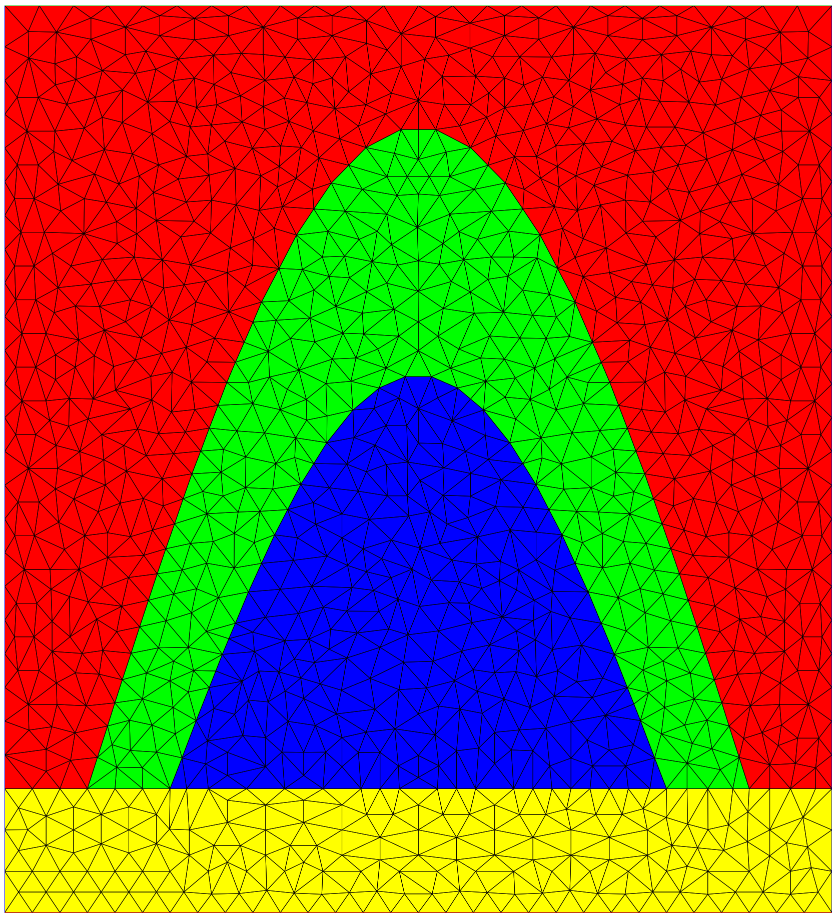


Figure 19: Coated pin grating determined by two parametric curves, Type 2.

– number n_{la} of layers above multilayer system
– number n_{lb} of layers below multilayer system
– number n_{mld} of multilayers inbetween
– number n_{ld} of layers in each multilayer
– number n_{corn} of corner points in each polygonal approximation (interfaces=randomly generated polygon)
– number n_{rand} of standard Gaussian distributed random numbers needed to generate interfaces (automatically created if n_{rand}=0, otherwise $n_{rand} = n_{corn}(n_{la}+n_{ld}n_{mld}+n_{lb})$)
– (optional) standard Gaussian distributed random numbers
– n_{la} widths of layers above multilayer system
– n_{ld} widths of layers in each multilayer
– n_{lb} widths of layers below multilayer system
– standard deviation $\sigma[i]$ of each layer interface for $i = 0, \ldots, (n_{la} + n_{ld}n_{mld} + n_{lb} - 1)$
– correlation length $corl[i]$ of each layer interface, for $i = 0, \ldots, (n_{la} + n_{ld}n_{mld} + n_{lb} - 1)$

Note that the first of the subsequent refractive indices of the grating must be that of the superstrate resp. that of the adjacent upper layer. If the width of the additional layers is positive, then the last of the subsequent refractive indices of the grating must be that of

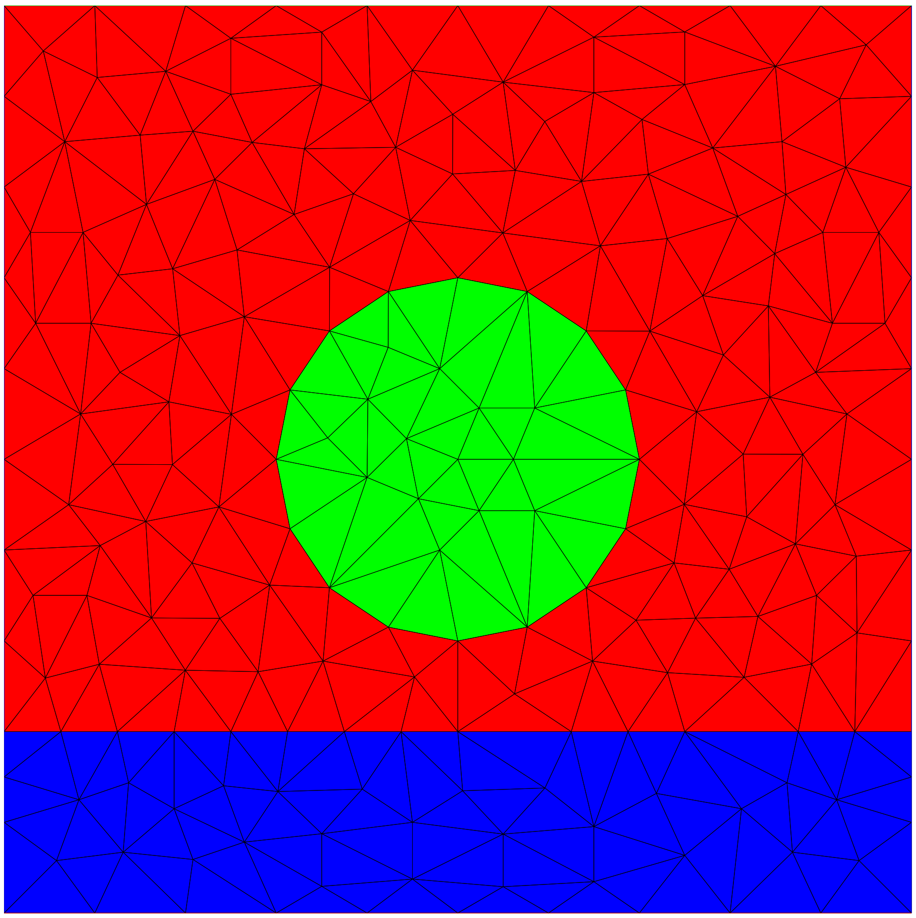


Figure 20: Box grating with two curves.

the substrate resp. that of the adjacent lower layer.

```
# Grating data:
  rough_mls k name
```

indicates a Multilayer System With Rough Interfaces k Times. This is just like above. However, k random realizations are computed and the output values are replaced by the mean values over the k computations. Standard deviations are added, too.

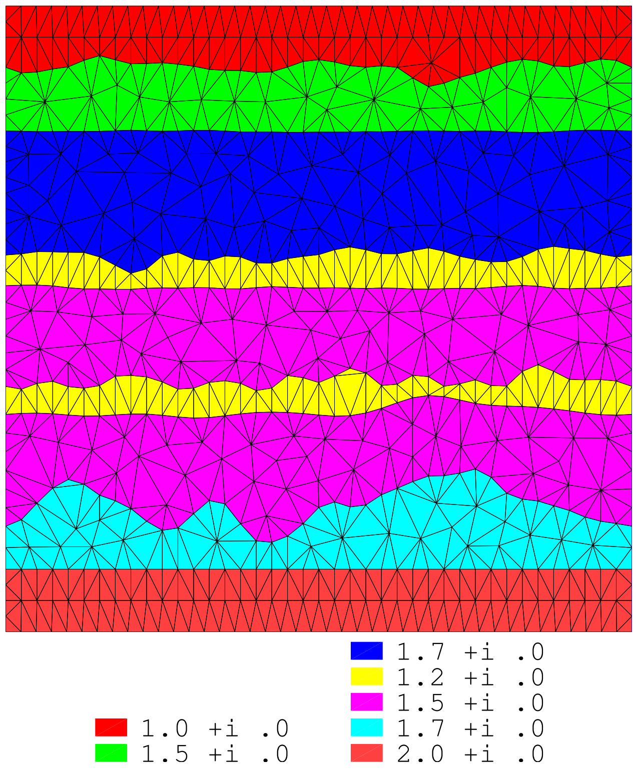


Figure 21: Rough multi-layer grating.

3.6 Stack grating by code words

For the stack grating, there appear the following code words in the geometry input of the name.dat file:

```
# Grating data:
stack k
line_1
line_2
    ...
line_2k
```

Here k is the number of profile curves in the stack. These profile curves are defined by the following $2k$ code word lines. Each profile curve is represented by two of the lines. They are listed from above to below. No intersection points of these curves are allowed. With the exception of pin curves the j-th curve ($j = 1, \dots, k$) takes the form $\{(f_x(j,t), f_y(j,t)) : 0 \le t \le 1\}$ with the first end point such that $f_x(j,0) = 0$, with the second end point such that $f_x(j,1) = 1$, and with $0 < f_x(j,t) < 1$ for $0 < t < 1$.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{echellea } \mathbf{R}\ d_1 \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get an Echelle Type A grating profile with right blaze angle greater than 45°, with depth d_1 and without coated layer (cf. Section 3.5). The profile is vertically shifted by h_1 μm. Note that $0 < d_1$ and d_1 is less than half the period d. If the input letter **R** is replaced by **L**, then the left interior angle is greater than 45°. Moreover, if the input letter **R** is replaced by **A** and the following number d_1 by α, then the left interior angle is α degrees.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{echelleb } \alpha \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get an Echelle Type B grating profile with angle α and without coated layer (cf. Section 3.5). The profile is vertically shifted by h_1 μm. Note that $0 < \alpha < 90$.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{echelle L } \alpha \text{ R } \beta \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get a General Echelle grating profile with a left blaze angle α (in degrees) and with a right blaze angle β (in degrees). Instead of the two inputs "L α" and "R β" one can choose also the inputs "A γ" for an apex angle γ (in degrees) or "D d_0" for a depth of the grating equal to d_0 (in μm). Moreover, any combination of two inputs of the types "A γ", "L α", "R β", and "D d_0" is accepted. However, the choice "A γ" and "D d_0" might be ambiguous. By definition it fixes an echelle grating with right blaze angle larger than the left. To get the flipped grating with left blaze angle larger than the right, the input should be "A γ D -d_0" (cf. the last section). The profile is vertically shifted by h_1 (in μm). Note that, for technical reasons, the blaze angles α and β must be less than 90°.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{binary } i\ d_0\ d_1\ d_2 \ldots d_{2i} \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get a Binary Grating Profile with i teeth, with height d_0, with transition points d_1, d_2, ..., d_{2i}, and without coating layer. In other words, before the shift, the grating function is zero between 0 and d_1, between d_2 and d_3, ... , between d_{2i-2} and d_{2i-1}, and between d_{2i} and the period d. The grating function is d_0 between d_1 and d_2, between d_3 and d_4, ... , between d_{2i-3} and d_{2i-2}, and between d_{2i-1} and d_{2i-1}. The profile is vertically shifted by h_1 μm. Note that $i \leq 6$, $0 < d_0$, $0 < d_1 < d_2 < \ldots < d_{2i} < d$ with d the period. For i equal to 1 or 2, either $d_1 = 0$ or $d_{2i} = d$ is allowed.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{trapezoid } d_1\ d_2\ d_3\ \alpha \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get a symmetric Trapezoidal Grating profile (cf. Section 3.5) with the trapezoid starting at $x = d_1$, ending at $x = d_2$, and with the angle α and the height d_3. The profile is vertically shifted by h_1 μm. Note that we require $d_3 > 0$, $0 \leq d_1 < d_2 \leq d$, and $0 < \alpha < 90$ with d the period. The height must be sufficiently small such that the trapezoid does not degenerate.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{profile } ccode \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get a profile curve Determined By A Smooth Simple Curve defined by $f_x(j,t) = t$ and $f_y(j,t) = ccode$. The profile is vertically shifted by h_1 μm. Note that $ccode$ is an expression of the parameter argument t, $0 \leq t \leq 1$, written in the c programming language. This expression will appear in the code as "$fct = ccode$;". Even an if case is possible. E.g. substituting the code "$0; if(t < 0.3) fct = t; else\ if(t < 0.6) fct = 0.6 - t; else\ fct = 0.$" into "$fct = ccode$;" leads to the meaningful code:

```
                       fct=0;
if          (t<0.3)    fct=t;
else   if   (t<0.6)    fct=0.6-t;
else                   fct=0.;
```

The code must be simple since the program can read no more than 399 symbols per input line.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{profile } ccode_1 \ /\#\#/\ ccode_2 \\ \text{line}_{2j} & = & h_1 \end{array}$$

then we get a profile curve Determined By A Smooth Simple Parametric Curve defined by $f_x(j,t) = ccode_1$ and $f_y(j,t) = ccode_2$. The profile is vertically shifted by h_1 μm. The same remarks as in the previous mode apply.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{profile}i\ ccode \\ \text{line}_{2j} & = & h_1\ d_1\ d_2\ \ldots\ d_i \end{array}$$

then we get a profile curve Determined By A Non-smooth Simple Curve defined by $f_x(j,t) = t$ and $f_y(j,t) = ccode$. The curve has i $(1 \le i \le 9)$ corners with the parameter arguments d_1, d_2, ..., d_i such that $0 < d_1 < d_2 < \ldots < d_i$. The profile is vertically shifted by h_1 μm. The same remarks as in the previous mode apply.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{profile}i\ ccode_1\ /\#\#/\ ccode_2 \\ \text{line}_{2j} & = & h_1\ d_1\ d_2\ \ldots\ d_i \end{array}$$

then we get a profile curve Determined By A Non-smooth Simple Parametric Curve defined by $f_x(j,t) = ccode_1$ and $f_y(j,t) = ccode_2$. The curve has i $(1 \le i \le 9)$ corners with the parameter arguments d_1, d_2, ... ,d_i such that $0 < d_1 < d_2 < \ldots < d_i$. The profile is vertically shifted by h_1 μm. The same remarks as in the previous mode apply.

Beside the above profile curves, pin curves are possible. Then the meaning of f_x and f_y is changed. The curve $t \mapsto (f_x(j,t), f_y(j,t))$ with $0 \le t \le 1$ connects the points $(f_x(j,0), f_y(j,0)) = (0,0)$ and $(f_x(j,1), f_y(j,1)) = (1,0)$. The corresponding pin curve is just the affine image of the last curve connecting the two points $(f_x(j_2,p_1), f_y(j_2,p_1))$ and $(f_x(j_2,p_2), f_y(j_2,p_2))$ of the profile curve with index j_2. I.e.:

$$\begin{aligned} t \ \mapsto \ \Big(& f_x(j_2,p_1) \\ & + f_x(j,t) * [f_x(j_2,p_2) - f_x(j_2,p_1)] \\ & - f_y(j,t) * [f_y(j_2,p_2) - f_y(j_2,p_1)], \\ & f_y(j_2,p_1) \\ & + f_x(j,t) * [f_y(j_2,p_2) - f_y(j_2,p_1)] \\ & + f_y(j,t) * [f_x(j_2,p_2) - f_x(j_2,p_1)] \Big) \end{aligned}$$

Thus, for each of the above profile curves, we can define several areas with new material attached to it and bounded by the just mentioned pin curves. These areas are listed immediately before the profile curve and in correspondence with their attachment from the right to the left. In other words, the index j_2 of the profile curve to which the j-th pin curve is attached to is the smallest integer l larger than j such that l is an ordinary profile curve, and, in case of two and more pins, the j-th pin curve is located to the right of the $(j+1)$-th pin curve. No intersections of pin and profile curves are allowed.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{pin}\ ccode \\ \text{line}_{2j} & = & p_1\ p_2 \end{array}$$

then we get a Simple Smooth Pin Curve defined by $f_x(j,t) = t$ and $f_y(j,t) = ccode$. The parameter arguments p_1 and p_2 of the connection points to the ordinary profile curve are fixed by the second line. The remarks on the profile curves apply also here.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{pin } \mathit{ccode}_1 \ /\#\#/ \ \mathit{ccode}_2 \\ \text{line}_{2j} & = & p_1 \ p_2 \end{array}$$

then we get a Simple Smooth Parametric Pin Curve defined by $f_x(j,t) = \mathit{ccode}_1$ and $f_y(j,t) = \mathit{ccode}_2$. The parameter arguments p_1 and p_2 of the connection points to the ordinary profile curve are fixed by the second line. The remarks on the profile curves apply also here.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{pin}i \ \mathit{ccode} \\ \text{line}_{2j} & = & p_1 \ p_2 \ d_1 \ d_2 \ \ldots \ d_i \end{array}$$

then we get a Simple Non-smooth Pin Curve defined by $f_x(j,t) = t$ and $f_y(j,t) = \mathit{ccode}$. The curve has i $(1 \le i \le 9)$ corners with the parameter arguments d_1, d_2, ..., d_i such that $0 < d_1 < d_2 < \ldots < d_i$. The remarks on the profile curves apply also here.

If the defining two lines are

$$\begin{array}{lcl} \text{line}_{2j-1} & & \text{pin}i \ \mathit{ccode}_1 \ /\#\#/ \ \mathit{ccode}_2 \\ \text{line}_{2j} & = & p_1 \ p_2 \ d_1 \ d_2 \ \ldots \ d_i \end{array}$$

then we get a Simple Non-smooth Parametric Pin Curve defined by $f_x(j,t) = \mathit{ccode}_1$ and $f_y(j,t) = \mathit{ccode}_2$. The curve has i $(1 \le i \le 9)$ corners with the parameter arguments d_1, d_2, ..., d_i such that $0 < d_1 < d_2 < \ldots < d_i$. The remarks on the profile curves apply also here.

For example, Figure 22 presents the stack grating generated by

```
# Grating data:
  stack 5
  echellea R 0.3
  1.2
  profile 0.3 * sin(2. * M_PI * t)
  0.8
  pin 0.5 − 0.5 * cos(M_PI * t) /##/ sin(M_PI * t)
  0.6 0.9
  pin1 0; if (t < 0.5) fct = t; else fct = 1. − t
  0.1 0.4 0.5
  profile t /##/ 0.
  0.
```

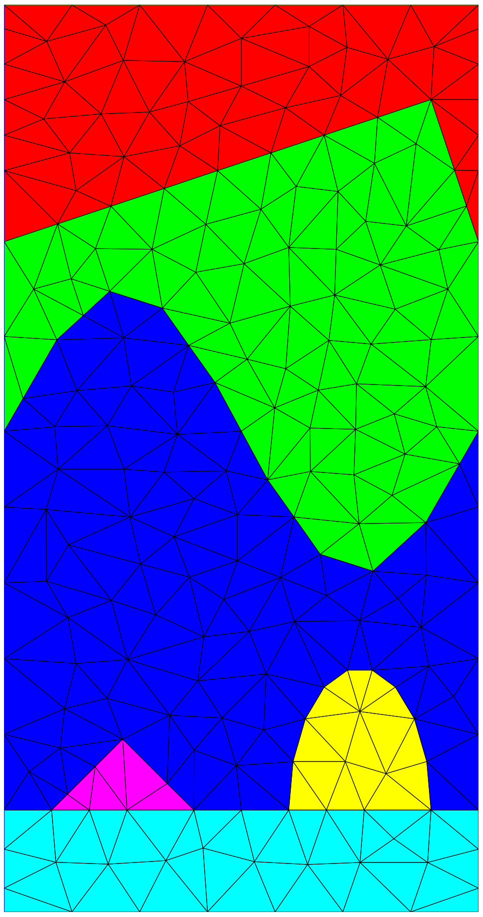


Figure 22: Stack grating.

4 Input of Refractive Indices

The optical properties of the materials involved in the grating are characterized by the refractive indices. Hence, for each material piece, the corresponding index must be added through the input file “name.dat”. This is done in lines like the following:

```
# Optical index (refractive index) of cover material.
  Air
# Optical indices of the materials of the upper coatings.
  1.1
  1.2
# Optical indices of the materials of the lower coatings.
  2.3 +i .0
  2.2 +i .5
  2.1 +i .0
# Optical index of substrate material.
  Al

  ...
# Wave length in micro m (lambda).
  .635

  ...
# Temperature:
  20.

  ...
# Optical indices of grating materials.
  1.2
  user
  1.7 +i .0
  2.3 +i .0
```

As seen in this example, the indices can be added as real or complex numbers (e.g. “1.1” resp. “2.2 +i .5”) or as code words of known materials (e.g. “Al” for aluminum). In the last case there is a program which interpolates the refractive index from a table in dependence on the temperature and on the wave length. The temperature enters only through such materials given by code words. If the indices are all numbers, then the temperature is not used. The code words for materials can be “Air”, “Ag”, “Al”, “Au”, “CsBr”, “Cu”, “InP”, “MgF2”, “NaCl”, “PMMA”, “PSKL”, “SF5”, “Si”, “TlBr”, “TlCl”, “Cr”, “ZnS”, “Ge”, “TiO2r”, “Quarz”, “AddOn”, and “Si1.0” - “Si2.0”. Here “Sia.b” with the real number “x=a.b” indicates a blending of “SiO” and “SiO_2” with the refractive index $n = (2 - x) \cdot n_{SiO} + (x - 1) \cdot n_{SiO_2}$. For example, “Air” corresponds to an index $n = 1$. Additionally, if the user has linked the right sources, the refractive indices of the program package `IMD` can be used as well. These are:

a-Al2O3_hagemann	a-Al2O3_llnl_cxro	a-Al2O3
a-Al2O3_palik	a-C_hagemann	a-C_llnl_cxro
Ac_llnl_cxro	a-C	Ac
a-C_palik	a-C_windt88	a-C_windt91
a-C_windt	Ag2O_llnl_cxro	Ag2O
Ag_hagemann	Ag_leveque	Ag_llnl_cxro
IMD-Ag	Ag_palik	Ag_windt
Al2O3-e	Al2O3-e_palik	Al2O3
Al2O3_palik	Al.3Ga.7As	Al.3Ga.7As_palik
AlAs	AlAs_palik	Al_llnl_cxro
IMD-Al	AlN_llnl_cxro	AlN
Al_palik	AlSb	AlSb_palik
Al_shiles	Al_windt	a-SiC_kortright
a-SiC_llnl_cxro	a-SiC	a-SiH
a-SiH_palik	a-Si_llnl_cxro	a-Si
a-SiO2_llnl_cxro	a-SiO2	a-SiO2_palik
a-Si_palik	a-Si_windt88	a-Si_windt91
a-Si_windt92	Au_canfield	Au_hagemann
Au_llnl_cxro	Au_nilsson	IMD-Au
Au_palik	Au_weaver	Au_windt
B2O3_llnl_cxro	B2O3	B4C_llnl_cxro
B4C	BaTiO3-e	BaTiO3-e_palik
BaTiO3	BaTiO3_palik	Be_llnl_cxro
Be	BeO_llnl_cxro	BeO
BeO_palik	Be_palik	B_llnl_cxro
B	BN_llnl_cxro	BN
C10H8O4_llnl_cxro	C10H8O4	C5H8O2_llnl_cxro
C5H8O2	CaF2_llnl_cxro	CaF2
CaF2_palik	CH2CCl2_llnl_cxro	CH2CCl2
CH2_llnl_cxro	CH2	Co2O3_llnl_cxro
Co2O3	Co3O4_llnl_cxro	Co3O4
Co_llnl_cxro	Co	CoO_llnl_cxro
CoO	Co_palik	CoSi2_llnl_cxro
CoSi2	Cr2O3_llnl_cxro	Cr2O3
Cr3C2_llnl_cxro	Cr3C2	Cr3C2_windt
Cr_llnl_cxro	IMD-Cr	CrO_llnl_cxro
Cr_palik	CsI_llnl_cxro	CsI
CsI_palik	Cu2O_llnl_cxro	Cu2O
Cu2O_palik	Cu4Si_llnl_cxro	Cu4Si
Cu_llnl_cxro	IMD-Cu	CuO_llnl_cxro
CuO	CuO_palik	Cu_palik
c-ZnSe	c-ZnSe_palik	c-ZnS
c-ZnS_palik	d-C_llnl_cxro	d-C
d-C_palik	d-C_windt	Fe2O3_llnl_cxro

Fe2O3	Fe3O4_llnl_cxro	Fe3O4
Fe_llnl_cxro	Fe	FeO_llnl_cxro
FeO	Fe_palik	GaAs_llnl_cxro
GaAs	GaAs_palik	GaAs_windt
GaP_llnl_cxro	GaP	GaP_palik
GaSb_llnl_cxro	GaSb	GaSb_palik
g-C_llnl_cxro	g-C	Ge_llnl_cxro
Ge	Ge_palik	H2O_llnl_cxro
H2O	H2O_palik	Hf_llnl_cxro
Hf	HfO2_llnl_cxro	HfO2
Hf-te_lynch	Hf-tm_lynch	Hf_windt
Hg_llnl_cxro	Hg	Hg_palik
HgTe	HgTe_palik	h-ZnS-e
h-ZnS-e_palik	h-ZnS	h-ZnS_palik
InAs	InAs_palik	IMD-InP
InP_palik	InSb	InSb_palik
Ir_llnl_cxro	Ir	Ir_palik
Ir_windt	ir-ZnSe	ir-ZnSe_palik
ir-ZnS	ir-ZnS_palik	KBr
KBr_palik	KCl	KCl_palik
K	K_palik	LiF
LiF_palik	LiNbO3-e	LiNbO3-e_palik
LiNbO3	LiNbO3_palik	Li
Li_palik	MgF2-e	MgF2-e_palik
IMD-MgF2	MgF2_palik	Mg_llnl_cxro
Mg	MgO_llnl_cxro	MgO
MgO_palik	Mn3O4_llnl_cxro	Mn3O4
Mn_llnl_cxro	Mn	MnO2_llnl_cxro
MnO2	Mo2C_llnl_cxro	Mo2C
MoC_llnl_cxro	MoC	Mo_llnl_cxro
Mo	MoO2_llnl_cxro	MoO2
MoO3_llnl_cxro	MoO3	Mo_palik
MoS2_llnl_cxro	MoS2	MoSi2_llnl_cxro
MoSi2	MoSi2_windt	Mo_windt88
Mo_windt91	Mo_windt92	IMD-NaCl
NaCl_palik	NaF	NaF_palik
Na	Na_palik	Nb2O5_llnl_cxro
Nb2O5	Nb_llnl_cxro	Nb
NbO2_llnl_cxro	NbO2	NbO_llnl_cxro
NbO	Nb_palik	Nb_weaver
Nb_windt	Ni.8Cr.2_llnl_cxro	Ni.8Cr.2
Ni.93V.07_llnl_cxro	Ni.93V.07	Ni_llnl_cxro
Ni	NiO_llnl_cxro	NiO
Ni_palik	Os_llnl_cxro	Os_lynch

Os	OsO2_llnl_cxro	OsO2
Os_palik	Os_windt	PbSe
PbSe_palik	PbS	PbS_palik
PbTe	PbTe_palik	Pd_llnl_cxro
Pd	PdO_llnl_cxro	PdO
Pd_palik	Pd_weaver	Pd_windt
Pt_llnl_cxro	Pt	PtO2_llnl_cxro
PtO2	PtO_llnl_cxro	PtO
Pt_palik	Pt_weaver	Pt_windt
Re2O7_llnl_cxro	Re2O7	Re_llnl_cxro
Re	ReO2_llnl_cxro	ReO2
ReO3_llnl_cxro	ReO3	Re-te_lynch
Re-tm_lynch	Re_windt	Rh2O3_llnl_cxro
Rh2O3	Rh_llnl_cxro	Rh
Rh_palik	Rh_weaver	Rh_windt
Ru_cox	Ru_llnl_cxro	Ru
RuO2_llnl_cxro	RuO2	RuO4_llnl_cxro
RuO4	RuSi_llnl_cxro	RuSi
Ru-te_weaver	Ru-tm_weaver	Ru_weaver
Ru_windt88	Ru_windt92	Sc2O3_llnl_cxro
Sc2O3	Sc_llnl_cxro	Sc
ScN_llnl_cxro	ScN	Se-e
Se-e_palik	Se	Se_palik
Si.25Ge.75	Si.25Ge.75_palik1	Si.25Ge.75_palik2
Si3N4_llnl_cxro	Si3N4	Si3N4_palik
Si3N4_windt	Si.5Ge.5	Si.5Ge.5_palik1
Si.5Ge.5_palik2	Si.8Ge.2	Si.8Ge.2_palik1
Si.8Ge.2_palik2	SiC_llnl_cxro	SiC
SiC_osantowski	SiC_palik	SiC_windt
SiC_yanagihara	Si_llnl_cxro	IMD-Si
SiO2-e	SiO2-e_palik	SiO2_llnl_cxro
SiO2	SiO2_palik	SiO_llnl_cxro
SiO	SiO_palik	Si_palik
Si_windt	Sn_llnl_cxro	Sn
SnO2_llnl_cxro	SnO2	SnO_llnl_cxro
SnO	SnTe	SnTe_palik
SrTiO3	SrTiO3_palik	Ta2O5_llnl_cxro
Ta2O5	TaC_llnl_cxro	TaC
Ta_llnl_cxro	Ta	TaN_llnl_cxro
TaN	Ta_palik	Ta_weaver
Ta_windt	Te-e	Te-e_palik
Te	Te_palik	ThF4
ThF4_palik	TiC_llnl_cxro	TiC
TiC_palik	Ti_kihara	Ti_llnl_cxro

Ti	TiN_llnl_cxro	TiN
TiN_palik	TiO2-e	TiO2-e_palik
TiO2_llnl_cxro	TiO2	TiO2_palik
Ti_weaver	Ti_windt	V2O3_llnl_cxro
V2O3	V2O5_llnl_cxro	V2O5
VC_llnl_cxro	VC	VC_palik
V_kihara	V_llnl_cxro	V
VN_llnl_cxro	VN	VN_palik
VO2_llnl_cxro	VO2	VO_llnl_cxro
VO	V_palik	V_weaver
W_llnl_cxro	W	WO2_llnl_cxro
WO2	WO3_llnl_cxro	WO3
W_palik	WSi2_llnl_cxro	WSi2
W_weaver	W_windt88	W_windt91
Y2O3_llnl_cxro	Y2O3	Y2O3_palik
Y_llnl_cxro	Y	Y_windt
Zn_llnl_cxro	Zn	ZnO_llnl_cxro
ZnO	ZnTe	ZnTe_palik
Zr_llnl_cxro	Zr_lynch	Zr
ZrN_llnl_cxro	ZrN	ZrO2_llnl_cxro
ZrO2	Zr-te_lynch	Zr_windt

On the other hand, the value of the refractive index can be interpolated from a user defined table, indicated by the name of the file. This file is to be located in the current directory of the computation (`CLASSICAL/CONICAL/OPTIM`). Its name must begin with the letter "u", must not have a "∼" as a second letter, and may consist of no more than five letters like e.g. "user". The file consists of at most 1000 lines each with three real numbers, the first is the wave length in micro meter, the second the real part of the corresponding optical index, and the third the imaginary part of the index. At the end of each line a comment beginning with the sign # can be added. Also the lines beginning with # are comments. Optical indices with negative real or imaginary parts are not admitted.

If the refractive index is given by a number, then its value is independent for all computations invoked by the input file. However, if the wavelength is varying in accordance with the input file, then a refractive index independent of the wavelength is not realistic. Hence, for varying wavelength, the input of refractive indices through user defined tables resp. through the above mentioned code words is mandatory. In this case an input by numbers is not accepted.

As seen in the example presented at the beginning of this section, first the index of the cover material is given. Then the indices of the materials of upper coated layers follow. These are rectangular layers over the whole period, and their number and widths are given in extra lines before the indices not presented in the example lines from above. If the number of coated layers is zero, then no lines with optical indices are needed. Next the indices of the materials of lower coated layers and that of the substrate follow. The indices of the materials in the area between upper and lower coatings (resp. between cover and substrate material if no rectangular coatings exist) are the last refractive indices of the input files. These indices are listed from above to below if possible. In some cases the ordering is indicated in the description of the geometrical part, or the indices have to be in accordance with the numbering of the material parts in the file "name.inp". In any case, the first index of the "grating materials" is to be the same as that of the adjacent last upper coating layer (resp. of the cover if there does not exist any rectangular upper coating) and the last index of the "grating materials" is to be the same as that of the adjacent first lower coating layer (resp. of the substrate if there does not exist any rectangular lower coating).

The input of refractive indices can be checked using the executables `FEM_CHECK` resp. `GFEM_CHECK` (cf. Sect. 5.3).

5 Computation of Efficiencies Using FEM in CLASSICAL

5.1 How to get an input file "name2.dat"?

First an input file "name2.dat" (cf. the enclosed data file in 12.2) in the subdirectory `CLASSICAL` is needed. To get this, change the directory to `CLASSICAL`, copy one of the existing files with tag ".dat", e.g. the file "example.dat" and call it "name2.dat".

```
cd DIPOG-2.1/CLASSICAL
cp example.dat name2.dat
```

Change "name2.dat" in the editor according to your requirements. You will find the necessary information as comments in the file "name2.dat". Indeed, each line beginning with "#" is a comment. Comment lines can be added and deleted without any trouble.

5.2 Simple calculation with minimal output

Now enter the command:

```
FEM name2
```

The program is running and produces an output on the screen. Additionally, a result file is produced (compare the similar file enclosed in point 12.6) the name of which is announced on the screen. You will find all Rayleigh coefficients, the efficiencies, and energies on both the screen and in this file. Note that the result file has the tag ".res" and is located in the subdirectory `RESULTS`. If a lot of data is produced, then computer programs should have an easy access to the data. To enhance readability by computer, a second output file can be produced setting a switch in "name2.dat" to yes. The name of this second file is the same as that of the first but with tag ".erg" instead of tag ".res". The file is normally located in the subdirectory `RESULTS`. The name "name3" of the result files "`../RESULTS/name3.res`" resp. "`../RESULTS/name3.erg`" is indicated by the "name2.dat" lines:

```
# Name of output file.
  name3
```

However, if the code is started from a directory different from `CLASSICAL` or if the output file should be written into a different directory, then the file is to be specified by adding its path as:

```
# Name of output file.
  path/name3
```

Again the tag ".res" resp. ".erg" will be added. In particular, for an output file in the current working directory use:

```
# Name of output file.
  ./name3
```

The computation proceeds over several levels, where the mesh size is halved at each new level. The maximal number of levels is indicated at the end of "name2.dat" e.g. by:

```
# Number of levels.
  3
```

However, if efficiencies are computed for more than one angle of incidence or for several wave lengths (or if a single value of angle/wave length is given by incremental or vector mode, i.e. beginning with the letter I or V), then the computation is performed for the highest level, only.

The ideal choice for the level would be the minimal positive integer such that the solution falls under a certain error bound. We have implemented the following choice. If "name2.dat" contains

```
# Number of levels.
  e ε
```

with the lower case letter "e" and with ε a number greater than zero, then the code computes the efficiency for the levels 1, 2, . . . (but no more than 15) until the maximum of the differences of efficiencies corresponding to two consecutive levels is less than ε. Thus, assuming a monotonic convergence, the smallest level for the given error bound ε is the smaller one of the last two consecutive levels. The efficiencies will be presented on the screen and in the output files for this level. (In other words, a computation for a level higher by one than that of the output is necessary for this variant.) If a computation over several angles, wave lengths or polarizations is required, then the "optimal" level will be determined for the first angle, for the first wave length resp. for the TM polarization, only. All other calculations are performed with this level. Clearly, there is no warranty that the efficiencies really deviate by a number less than ε from the true values.

5.3 Check before computation, more infos, and plots?

Instead, if `openGL` is available and if you wish to check your input data, then use the command:

```
FEM_CHECK name2
```

All the input information without output data will appear on the screen and in the result file. Moreover, there will appear a picture of the grating geometry with indicated refractive indices on the screen. The picture looks like that on the right-hand side in Figure 5. If

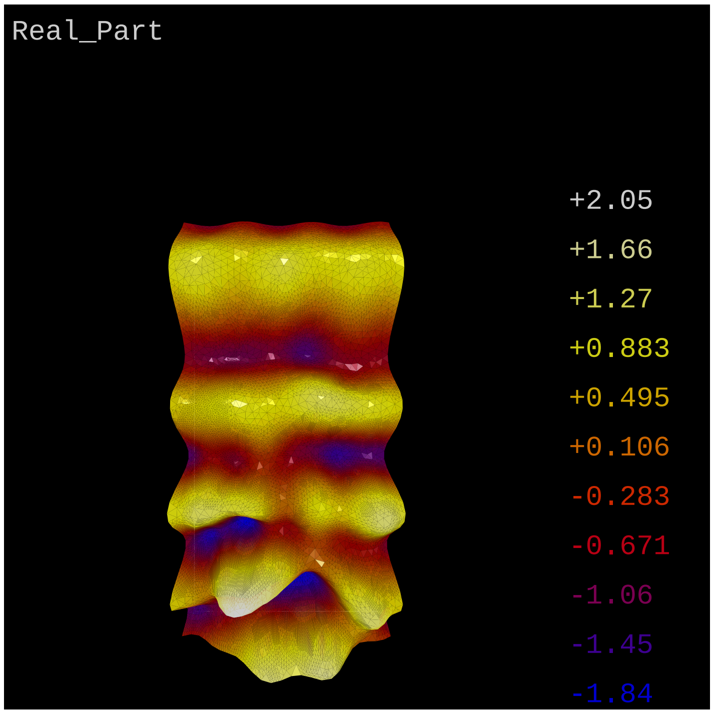


Figure 23: Real part of transverse component of magnetic field. Output of `FEM_PLOT` via `openGL` in `CLASSICAL`.

indicated in the data file "name.dat", then an eps.file of the cross-section is produced.

Instead, if you use the command

`FEM_FULLINFO` name2

then the same is done as in point 5.2. Additionally, there appears more information (including the full input data and the convergence history, cf. the enclosed file in point 12.5) on the screen and in the result file.

Instead, if `openGL` or `GNUPLOT` is available and if you use the command

`FEM_PLOT` name2

then you have the same results as in point 5.2. Additionally, you will see pictures of the real part, the imaginary part (cf. Figure 23 for an `openGL` picture and Figure 24 for a `GNUPLOT` picture), and the square modulus of the solution (z-component of electric field for TE polarization, z-component of magnetic field for TM polarization). Note that the square modulus is proportional to the energy intensity distribution of the wave. Moreover, similar pictures for the fields above and below the coated grating area will be plot. Program stops at each picture.

To control the graphical facilities of `GLTOOLS` in `FEM_PLOT`, use:

Backspace: Enter user control mode.
tab: toggle state change mode.
Return: Quit user control mode.
Space: Mode control.
+: Increase mouse sensitivity.
,: decrease control parameter.

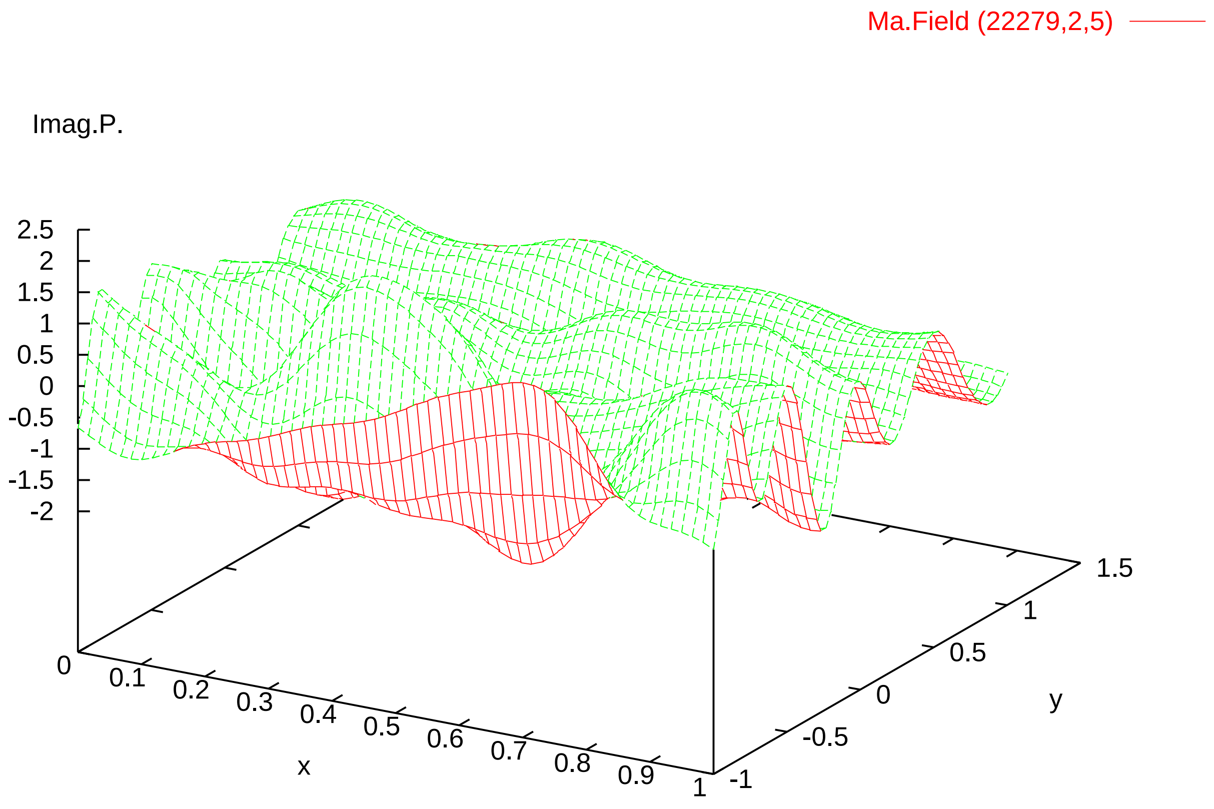


Figure 24: Imaginary part of transverse component of magnetic field. Output of executable `GFEM_PLOT` in `CLASSICAL` via `GNUPLOT`.

-: Decrease mouse sensitivity.
.: increase control parameter.
¡: Zoom out.
¿: Zoom in.
?: This help.
B: Toggle background color (black/white).
d: Dump actual picture to ppm file (look for *-*.ppm).
F: Toggle rendering volume frame (bounding box) drawing.
I: Change number of isolines.
.: increase control parameter by a factor.
O: Toggle Ortho.
D: Print actual picture using ppm dump.
R: Reset to internal default.
S: Save actual state (look for .*-rndstate).
V: Start/Stop video recording.
a: Switch to GUI.
c: Toggle remembered lists.
g: Toggle Gouraud/flat shading.
h: This help.

i: Toggle isoline mode.
,: decrease control parameter by a factor.
l: Toggle level surface mode.
m: Toggle model display when moving.
p: Dump actual picture to eps file (look for *-*.eps).
q: Mode control (Quit).
r: Restore last saved state.
v: Toggle vscale for plane sections.
w: toggle wireframe mode.
x: Show x orthogonal plane section.
y: Show y orthogonal plane section.
z: Show z orthogonal plane section.
prev: toggle state change mode.
next: toggle state change mode.
left: move left.
up: move up.
right: move right.
down: move down.
Backspace: Enter user control mode.

To continue the computation of the main program press Bar/Space.

After the installation with the package `GNUPLOT`, the pictures are created interactively with the program on the main terminal window. To continue the computation of the main program, click the main terminal window and press Enter/Return.

Further, if you use the command

```
GFEM_MATLAB name2
```

then you have the same computational results as in point 5.2. Additionally, you will obtain data files prepared for a `Matlab` call. More precisely, using `GFEM_MATLAB` requires a data file "name2.dat" for a calculation over a single wavelength, a single angle of incidence, and a single level of discretization. Note that if the last should be larger than one, then a single level computation can be enforced by defining the angle of incidence with an incremental input (e.g., the input "I 45 46 20" for the angle results in a single computation with an angle of 45°). Depending on the input polarization state, the program produces the files:

"fct_RP_TE.m":	real part of $z-$component of electric field vector for TE polarization
"fct_IP_TE.m":	imaginary part of $z-$component of electric field vector for TE polarization
"fct_IN_TE.m":	intensity of electric field vector for TE polarization
"fct_RP_TM.m":	real part of $z-$component of magnetic field vector for TM polarization
"fct_IP_TM.m":	imaginary part of $z-$component of magnetic field vector for TM polarization
"fct_IN_TM.m":	intensity of electric field vector for TM polarization

Here the intensity I is defined as $I := \Re e\, n \cdot |E|^2$, where n is the refractive index and where

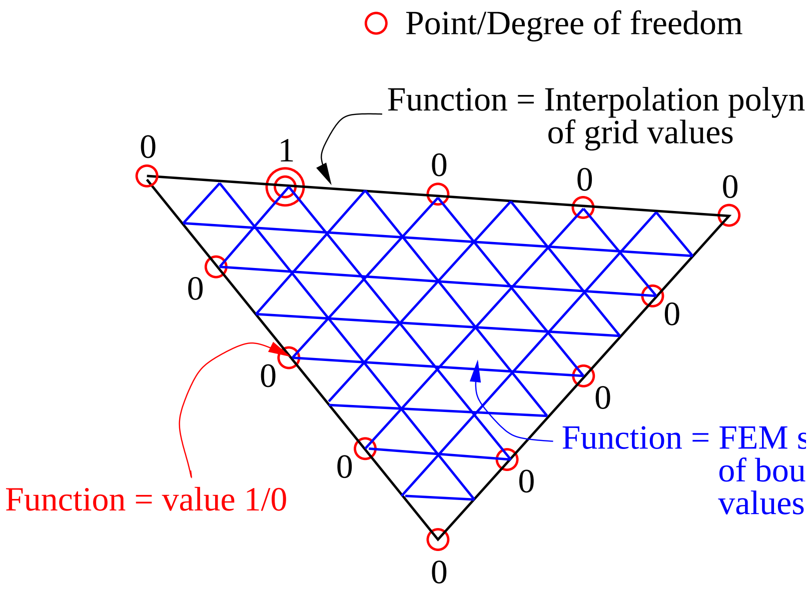


Figure 25: Trial basis function over a single grid triangle.

E is the electric field vector. If the package `Matlab` is installed at your computer, you can call these files from `Matlab` to see the plots.

Finally, if you use the command

```
GFEM_MOVIE name2
```

then the input file "name2.dat" is required to contain a single wave length and a single state of polarization, only. The program produces a `Matlab` file "fct_TE.m" for TE polarization and "fct_TM.m" for TM polarization. If this is called from `Matlab` a movie of the time dependent third component of the electric field for TE polarization and of the magnetic field for TM is shown. Moreover, a movie file "fct_TE.avi" resp. "fct_TM.avi" is created

6 Computation of Efficiencies Using GFEM in CLASSICAL

The same computation from the last section can be performed by generalized FEM (cf. the result file enclosed in point 12.6). The latter is nothing else than the variational approach of the conventional FEM combined with a new trial space for the approximation of the unknown solution. The trial space is defined over the triangular FEM partition and the trial functions are piecewise approximate solutions of the Helmholtz equation. More precisely, for integers n_{DOF} and n_{LFEM} with $0 < n_{\mathrm{DOF}}$, with $1 < n_{\mathrm{LFEM}}$, and with $[n_{\mathrm{LFEM}} + 1]$ a multiple of $[n_{\mathrm{DOF}} + 1]$, the degrees of freedom of the trial space are the function values at the corner points of the triangulations and at the n_{DOF} points of a uniform partition of each triangle side. The restrictions of the trial functions to the triangle sides are polynomial interpolants of the degrees of freedom. The restrictions of the trial functions to the triangles are the finite element solutions of the Dirichlet problem for the Helmholtz equation over a uniform triangulation of the partition triangle into $[n_{\mathrm{LFEM}}+1]\times[n_{\mathrm{LFEM}}+1]$ equal subtriangles (cf. the trial functions over the grid triangle indicated in Figure 25 with $n_{\text{DOF}} = 3$ and $n_{\text{LFEM}} = 7$). Hence, `GFEM` treats the same problems as `FEM` but is more convenient to treat higher wave numbers and faster for simple geometries. In order to accelerate the computation, one can use a FEM grid which is a uniform refinement of a coarse grid with each coarse triangle split into $[n_{\text{UPA}} \times n_{\text{UPA}}]$ congruent subtriangles. In this case, the trial space over the congruent grid triangles (approximate Dirichlet solutions) have to be computed only ones and can be reused several times.

To use generalized FEM for FEM, the executables `FEM`, `FEM_CHECK`, `FEM_FULLINFO`, and `FEM_PLOT` are to be replaced by the codes `GFEM`, `GFEM_CHECK`, `GFEM_FULLINFO`, and `GFEM_PLOT`, respectively. These executable use the same input file "name2.dat" but require the additional input file "generalized.Dat" (cf. the enclosed file in 12.3). Normally the latter is to be located in the current working directory. If there is no such file in the current working directory and if the output is written into a directory indicated by a certain path, then the code looks for the "generalized.Dat" file also in the directory determined by this path. The file "generalized.Dat" fixes the parameters:

n_{DOF}
: Additional degrees of freedom on each triangle side. Indeed, the trial functions on each subdivision triangle are approximate solutions of the Dirichlet problem for the Helmholtz equation s.t. their restriction to the triangle sides coincides with the Lagrange interpolation polynomials on the triangle side. Here interpolation is taken over a uniform grid with $[n_{\text{DOF}} + 2]$ interpolation knots including the two end-points.

n_{LFEM}
: Approximate solution determined by FEM over subdivision triangle, where an additional uniform FEM partition on each grid triangle is chosen such that the stepsize is side length divided by $[n_{\text{LFEM}} + 1]$.

n_{UPA}
: This is for additional uniform partition of all primary grid triangles into $n_{\text{UPA}} \times n_{\text{UPA}}$ equal subdomains, i.e. the original side of the grid triangle is split into n_{UPA} sides of uniform partition subtriangles. After this uniform refinement the degrees of freedom and the trial space of approximate Helmholtz solutions are defined using n_{DOF} and n_{LFEM}.

Change these parameters according to your diffraction problem, computer memory capacity, and computing time requirements. How should they be chosen?

Suppose that n_{GP} is the number of grid points. Roughly speaking: For the FEM, the computation time as well as the necessary storage capacity is proportional to $[n_{\text{GP}}]^2$. The time for `GFEM` is proportional to

$$\frac{[n_{\text{GP}}]^2 \times [n_{\text{LFEM}} + 1]^2}{[n_{\text{UPA}}]^2} + [n_{\text{GP}}]^2 \times [n_{\text{DOF}} + 1]^2 \tag{6.1}$$

and the storage to $[n_{\text{GP}}]^2 \times [n_{\text{DOF}} + 1]^2$. Hence, a doubling of $[n_{\text{LFEM}} + 1]$ leads to about the same computing time as halving the mesh size of the grid. Taking into account (6.1), we recommend to choose n_{UPA} as high as possible since the accuracy is almost independent of n_{UPA} but the computing time reduces significantly. The only exceptional case, when a larger n_{UPA} is not efficient, occurs if the geometry forces the triangulation to have a few large triangles and a huge number of small triangles (e.g. geometries with thin layers). In this case, the next level uniform partition increases the number of grid points and the computation time significantly, whereas a standard triangulation with halved mesh size leads to a small increase of grid points and to about the same numerical error.

If $n_{\mathrm{DOF}} = 0$ and $n_{\mathrm{LFEM}} = 1$, then the conventional FEM is computed. With $n_{\mathrm{DOF}} = 0$, a higher n_{LFEM} is not recommended. For $n_{\mathrm{DOF}} > 1$ and not so restrictive accuracy requirements, $n_{\mathrm{LFEM}} = 2n_{\mathrm{DOF}} + 1$ is a good choice. For $n_{\mathrm{DOF}} > 1$ and challenging accuracy requirements, a larger n_{LFEM} is useful. E.g. if $n_{\mathrm{DOF}} = 3$, then $n_{\mathrm{LFEM}} = 31$ is a good choice. For $n_{\mathrm{DOF}} = 7$, one should take, e.g., $n_{\mathrm{LFEM}} = 127$ and, for $n_{\mathrm{DOF}} = 15$, the value $n_{\mathrm{LFEM}} = 511$. However, large n_{LFEM} will lead to long computation times at least if n_{UPA} is not large. In case of large wave numbers, long computation times cannot be avoided. More hints on how to chose the right n_{DOF} and n_{LFEM} will be given in the numerical tests in Section 9.

If there is a system of many layers above or below the grating structure, the computation of the non-local boundary condition may become slow. On the other hand, a computation based on piecewise linear approximation like for the FEM matrix might be unnecessary. Note that each boundary segment is split into $n_{\mathrm{LFEM}} + 1$ equal parts, and the polynomial basis function is approximated by a piecewise linear interpolation over this uniform grid. In order to reduce computation time, the user can choose a smaller n_{LFEM} for the discretization of the boundary condition. To this end he has to set the environment variable `BND_n_LFEM` to the desired value `BND_`n_{LFEM}. This value should be such that `BND_`$n_{\mathrm{LFEM}} + 1$ is a multiple of the number $n_{\mathrm{DOF}} + 1$ and that $n_{\mathrm{DOF}} \leq$ `BND_`$n_{\mathrm{LFEM}} \leq n_{\mathrm{LFEM}}$.

For the computation of the approximate trial functions of the generalized finite element method, a finite element system of dimension $(n_{\mathrm{LFEM}} - 1) * n_{\mathrm{LFEM}}/2$ is to be solved. If the user sets the environment variable `CHOOSE_PMETHOD` to *yes*, then the solution of this sparse finite element system is reduced to the solution of an $(n_{\mathrm{DOF}} - 1) * n_{\mathrm{DOF}}/2$ dimensional finite element system, where the trial functions are the piecewise linear interpolations of the bubble functions for the p-version of the finite element method. The polynomial degree is $p = n_{\mathrm{DOF}} + 1$. Thus the generalized finite element method turns into a p-method with elimination of internal degrees of freedom and with finite element matrix approximated by piecewise linear interpolation.

7 Computation of Efficiencies Using FEM/GFEM in CONICAL

The computation in the case of conical diffraction is completely the same as for the classical computation (cf. the result file enclosed in point 12.7). The same names of executables can be used as in Section 2. The only differences are:

- All computations are to be done in `CONICAL` instead of `CLASSICAL`.
- Of course, the now used input file “name2.dat” (cf. the enclosed data file in 12.4) is longer than that of the classical case. To create such a file copy the example file “example.dat” in `CONICAL` not that in `CLASSICAL`.
- The input file for the generalized method “generalized.Dat” is the same but has a

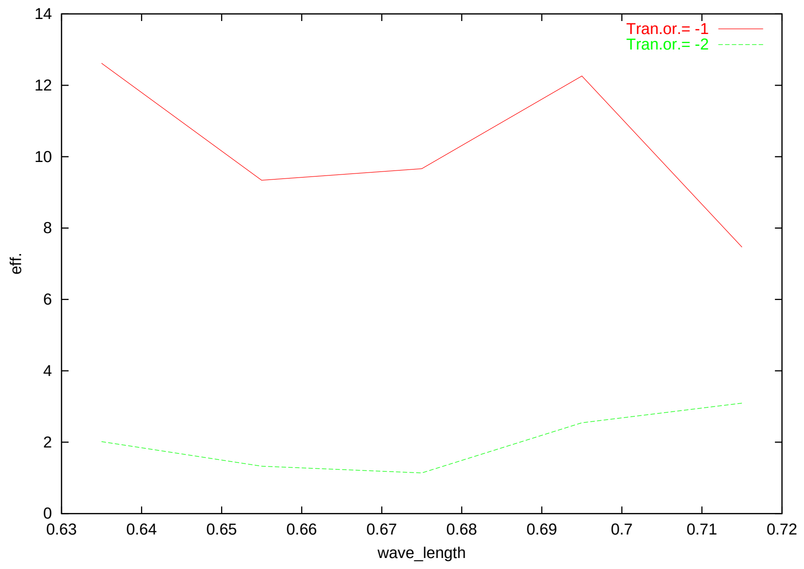


Figure 26: Efficiencies depending on wavelength. Output of `PLOT_PS`.

new name. Now it is called "conical.Dat".

8 Plot a Graph with the Efficiencies

If a result file "name3.res" is produced containing the values for several wave lengths or incident angles, then one can have a look at the two-dimensional graph of the efficiencies depending on the wave length or incident angle. Make sure to be in the subdirectory `RESULTS`, where the result file, e.g. "name3.res" exists. Then enter the command:

```
PLOT_DISPLAY name3 *1,*2,*3,*4
```

Here *1, *2, *3, and *3 stand for the efficiency/energy to be plot. E.g. setting *1 equal to R-1 means efficiency of reflected mode of order -1, setting *2 equal to T+0 means efficiency of transmitted mode of order 0, setting *3 equal to RE means total reflected energy, and setting *4 equal to TE means total transmitted energy. The number of efficiency/energy can vary between one and nine. Now a graph with the efficiencies/energies pops up on the screen (cf. Figure 26). To interrupt the presentation of the picture, press Enter/Return.

Alternatively, one can enter the command:

```
PLOT_PS name3 *1,*2,*3,*4
```

Everything is like in the previous case. However, instead of showing the graph on the screen, a ps file (cf. Figure 26) is produced. The name of the ps file will be printed on the screen.

In the case of classical diffraction and for result files with varying wave lengths or/and varying angles of incidence, one can enter the command:

```
PLOT_MATLAB name3 *1
```

Everything is like in the previous case. However, instead of creating a two-dimensional graph, a `Matlab` file is produced. This file with the tag ".m" can be called from `Matlab` and shows a graph or an isoline picture of the efficiency, phase shift, resp. energy depending on the wave length and the angles of incidence. The name of the `Matlab` file will be printed on the screen. Similarly, in the case of classical diffraction and for result files with varying wave lengths or/and varying angles of incidence, one can enter the command:

```
PLOT_GNUPLOT name3 *1
```

Everything is like in the previous case. However, instead of creating the `Matlab` file a gnuplot ps file is produced. The name of the ps file will be printed on the screen.

9 Parameter Test for GFEM

Here we present the results of test computations to give an orientation for the choice of the parameters n_{DOF} and n_{LFEM}. Recall that n_{GP} is the number of grid points. For simplicity, we present results with $n_{\mathrm{UPA}} = 1$, only. We consider a trapezoidal grating with basis angle of 60° and with one material which covers 60% of the period. The height of the trapezoid is 0.3 times length of period and the refractive index is 2.0. Moreover, we assume an additional layer which covers the whole period and which has a refractive index of 1.3 and a height of 0.05 times period. The substrate has a refractive index of 1.5 and the superstrate is air. The grating is illuminated in a classical TE scenario by light of wave length 635nm under an incidence angle of 65°. The length of one period is 1μm, 2μm, 4μm, 8μm, and 16μm. In other words, we have chosen the geometry generated by the code words

```
# Grating data:
  trapezoid 60 a 1 b c
```

where a is 0.6 times the length of the period, b is 0.3 times period, and c is 0.05 times period.

Now the accuracy and the best choice of parameters depend on the maximal relative wave number which is period times refractive index over wave length. We have checked the accuracy in percent. For instance, one percent accuracy means that: The absolute error of the percentage of energy reflected by the grating is less than 1% (the value itself is less than 100%). The absolute error of the percentage of light transmitted into the minus first order is less than 1% (the efficiency itself is less than 100%). The absolute error of real or imaginary part of the Rayleigh coefficient of the minus first reflected mode is less than 0.01 (the modulus of the coefficient itself is less than one).

The corresponding relative mesh size[16] h_r and the corresponding number of refinement levels (starting from 1 for the coarsest) necessary to achieve an accuracy up to 1%, 0.1%, and 0.01%, respectively, are presented in the Tables 1–3. Here we define the relative mesh

[16] Of course, the values presented in the Tables 1–3 are taken from a discrete set of test values, only. Indeed, we have computed only those relative mesh sizes which result from the halving the mesh size strategy realized in the code by switching to higher refinement levels.

size h_r of the grid by[17]

$$h_r = \frac{2\pi}{d} \frac{h}{n_{\mathrm{DOF}} + 1}$$

with h the absolute mesh size of the triangulation and with d the length of the period. By the symbol k_r in the tables we denote the maximal relative wave number (length of period d times refractive index divided by wave length). The numerical methods are either FEM or GFEM($n_{\mathrm{DOF}}, n_{\mathrm{LFEM}}$), i.e. the GFEM with the parameters n_{DOF}, n_{LFEM}, and $n_{\mathrm{UPA}} = 1$. Stars indicate that the accuracy is not reached due to the restricted main memory of the computer. The number of grid points n_{GP} is 67 for the first level, 75 for the second, 169 for the third, 600 for the fourth, 2 430 for the fifth, 8 858 for the sixth, 39 698 for the seventh, 159 140 for the eighth, and 637 914 for the ninth (cf. the computation time in (6.1)).

It is impossible to derive a general recommendation from the numbers in the Tables 1–3. We have indicated the necessary relative mesh size for the fastest[18] method with parameter $n_{\mathrm{UPA}} = 1$ by bold letters. However, the methods with doubled $[n_{\mathrm{DOF}} + 1]$ and $[n_{\mathrm{LFEM}} + 1]$ and doubled mesh size (one lower refinement level) require almost the same computation time and lead to the same accuracy. If $[n_{\mathrm{LFEM}} + 1]$ is large and the grid is of a higher refinement level, then the computing time can be reduced by first generating a preliminary grid with the doubled maximal mesh size and second applying an additional uniform refinement step of each triangle into four equal subtriangles. Recall that the trial functions for congruent triangles need to be computed only once. In other words, reducing the level by one and changing n_{UPA} from 1 to 2, turns GFEM into a competitive method even if $[n_{\mathrm{LFEM}} + 1]$ is large. Similarly, the level can be reduced by 2 or 3, and n_{UPA} can be set to 4 or 8. So GFEM with larger n_{DOF} and n_{LFEM} outperforms the GFEM indicated by bold letters.

For TM polarization and the same grating and light scenario, we get similar results. E.g., in the case of $k_r = 12.60$ (i.e. $d = 4$) and GFEM(3,31), we get an error of 1%, 0.1%, and 0.01% choosing the refinement levels 4, 6, and 6, respectively. For $k_r = 25.20$ (i.e. $d = 8$) and GFEM(7,127), we get an error of 1%, 0.1%, and 0.01% choosing the refinement levels 4, 4, and 5, respectively.

[17]Note that the mesh size shown on the screen or in the result files “name.res” after calling the program `FEM_FULLINFO` and `GFEM_FULLINFO` are just the h_r.

[18]Fastest method means the one with the smallest complexity estimate (6.1).

METHOD	k_r=3.15 (d=1)	k_r=6.30 (d=2)	k_r=12.60 (d=4)	k_r=25.20 (d=8)	k_r=50.39 (d=16)
FEM	**0.401** (4)	0.089 (6)	0.024 (8)	$\star\star\star$	$\star\star\star$
GFEM(1,3)	0.401 (3)	0.201 (4)	0.044 (6)	**0.012** (8)	$\star\star\star$
GFEM(1,7)	0.401 (3)	0.201 (4)	0.044 (6)	0.023 (7)	$\star\star\star$
GFEM(1,15)	0.401 (3)	0.201 (4)	0.044 (6)	0.023 (7)	0.012 (8)
GFEM(3,7)	0.524 (1)	0.201 (3)	0.044 (5)	0.012 (7)	$\star\star\star$
GFEM(3,15)	0.524 (1)	0.324 (2)	0.100 (4)	0.022 (6)	**0.012** (7)
GFEM(3,31)	0.524 (1)	0.324 (2)	0.100 (4)	0.044 (5)	0.012 (7)
GFEM(3,63)	0.524 (1)	0.324 (2)	0.100 (4)	0.044 (5)	0.044 (5)
GFEM(7,15)	0.262 (1)	**0.262** (1)	0.050 (4)	0.011 (6)	$\star\star\star$
GFEM(7,63)	0.262 (1)	0.262 (1)	0.100 (3)	0.050 (4)	0.022 (5)
GFEM(7,127)	0.262 (1)	0.262 (1)	0.100 (3)	0.050 (4)	0.022 (5)
GFEM(7,255)		0.262 (1)	0.100 (3)	0.100 (3)	0.022 (5)
GFEM(15,31)		0.131 (1)	**0.131** (1)	0.011 (5)	$\star\star\star$
GFEM(15,127)		0.131 (1)	0.131 (1)	0.050 (3)	0.011 (5)
GFEM(15,255)			0.131 (1)	0.050 (3)	0.025 (4)
GFEM(15,511)				0.081 (2)	0.025 (4)

Table 1: Relative mesh size h_r (refinement levels) necessary to reach 1% accuracy.

METHOD	k_r=3.15 (d=1)	k_r=6.30 (d=2)	k_r=12.60 (d=4)	k_r=25.20 (d=8)	k_r=50.39 (d=16)
FEM	0.047 (7)	0.024 (8)	$\star\star\star$	$\star\star\star$	$\star\star\star$
GFEM(1,3)	0.088 (5)	0.044 (6)	**0.012** (8)	$\star\star\star$	$\star\star\star$
GFEM(1,7)	0.201 (4)	0.088 (5)	0.023 (7)	$\star\star\star$	$\star\star\star$
GFEM(1,15)	0.201 (4)	0.088 (5)	0.023 (7)	0.012 (8)	$\star\star\star$
GFEM(3,7)	0.100 (4)	0.044 (5)	0.012 (7)	$\star\star\star$	$\star\star\star$
GFEM(3,15)	0.201 (3)	0.100 (4)	0.022 (6)	**0.012** (7)	$\star\star\star$
GFEM(3,31)	**0.524** (1)	0.201 (3)	0.044 (5)	0.022 (6)	$\star\star\star$
GFEM(3,63)	0.524 (1)	0.201 (3)	0.044 (5)	0.022 (6)	**0.012** (7)
GFEM(7,15)	0.100 (3)	**0.100** (3)	0.011 (6)	$\star\star\star$	$\star\star\star$
GFEM(7,63)	0.100 (3)	0.262 (1)	0.050 (4)	0.011 (6)	$\star\star\star$
GFEM(7,127)	0.262 (1)	0.262 (1)	0.100 (3)	0.050 (4)	0.011 (6)
GFEM(7,255)		0.262 (1)	0.100 (3)	0.050 (4)	0.022 (5)
GFEM(15,31)		0.050 (3)	0.011 (5)	$\star\star\star$	$\star\star\star$
GFEM(15,127)		0.131 (1)	0.050 (3)	0.025 (4)	$\star\star\star$
GFEM(15,255)			0.081 (2)	0.050 (3)	0.011 (5)
GFEM(15,511)				0.050 (3)	0.025 (4)

Table 2: Relative mesh size h_r (refinement levels) necessary to reach 0.1% accuracy.

METHOD	k_r=3.15 (d=1)	k_r=6.30 (d=2)	k_r=12.60 (d=4)	k_r=25.20 (d=8)	k_r=50.39 (d=16)
FEM	0.013 (9)	**0.013** (9)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(1,3)	0.023 (7)	0.023 (7)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(1,7)	0.044 (6)	0.044 (6)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(1,15)	0.088 (5)	0.044 (6)	**0.012** (8)	⋆⋆⋆	⋆⋆⋆
GFEM(3,7)	0.022 (6)	0.022 (6)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(3,15)	0.044 (5)	0.044 (5)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(3,31)	0.100 (4)	0.100 (4)	0.012 (7)	⋆⋆⋆	⋆⋆⋆
GFEM(3,63)	0.201 (3)	0.100 (4)	0.022 (6)	**0.012** (7)	⋆⋆⋆
GFEM(7,15)	0.022 (5)	0.022 (5)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(7,63)	0.100 (3)	0.100 (3)	0.011 (6)	⋆⋆⋆	⋆⋆⋆
GFEM(7,127)	**0.262** (1)	0.162 (2)	0.022 (5)	0.011 (6)	⋆⋆⋆
GFEM(7,255)			0.050 (4)	0.022 (5)	**0.011** (6)
GFEM(15,31)		0.025 (4)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
GFEM(15,127)		0.081 (2)	0.011 (5)	⋆⋆⋆	⋆⋆⋆
GFEM(15,255)			0.025 (4)	0.011 (5)	⋆⋆⋆
GFEM(15,511)				0.025 (4)	0.011 (5)

Table 3: Relative mesh size h_r (refinement levels) necessary to reach 0.01% accuracy.

10 Optimization Tools

10.1 Optimizing using `OPTIMIZE` in `OPTIM`

10.1.1 The optimization problem

Optimizing an optical grating means the following. First the user has to fix:

- the period of the grating
- the refractive indices of the cover and substrate material
- one or more wave lengths of light $\lambda_j,\ j = 1, \dots j_\lambda$
- one or more angles of incidence $\theta_k,\ k = 1, \dots, k_\theta$ and $\phi_l,\ l = 1, \dots, l_\phi$
- one or two types of polarization $t_m \in \{\text{TE}, \text{TM}\},\ m = 1, \dots, m_t \le 2$

Beside all these, he has to fix a class of gratings (cf. the subsequent Section 10.2). Such a class is a certain grating geometry made of several materials. The gratings of the class can be described by a small number of real parameters $r_i,\ i = 1, \dots, N$, which are either geometrical parameters like lengths, widths and heights or the real and/or imaginary parts of the refractive indices. Together with the class of gratings, the user fixes the number of parameters. He restricts the admissibility (feasibility) domain of real parameters by adding upper bounds u_i and lower bounds l_i. To each set of parameters $r := \{r_i\}$ with $l_i \le r_i \le u_i,\ i = 1, \dots, N$ there exists a unique grating. Searching for an optimal grating is equivalent to finding the optimal set of parameters r. The optimality of a grating resp. set of parameters, however, is measured by an objective functional $f(r)$ which is an arithmetic expression of the following data of the grating determined by the parameter set r:

- efficiencies $e_n^\pm$
- phase shifts[19] $p_n^\pm$ (in degrees)
- reflected energy $e^+ := \sum_n e_n^+$ (in per cent)
- transmitted energy $e^- := \sum_n e_n^-$
- total energy $e := e^+ + e^-$

Here n runs through the indices of reflected resp. transmitted plane wave modes. Note that all these efficiencies, phase shifts, and energies depend of course on λ_j, θ_k, ϕ_l, and t_m. Moreover, for the conical diffraction, there exist two different efficiencies ($e_n^{1,\pm}$ and $e^{2,\pm}$) and phase shifts ($p_n^{1,\pm}$ and $p^{2,\pm}$) depending on the polarization splitting (cf. Section 2.3). The optimization problem

$$\begin{array}{rcl} f(r) & \longrightarrow & \inf \\ r \in \text{class}: & & \\ l_i \le r_i \le u_i & & \end{array} \qquad (10.1)$$

[19] In the case of TE polarization, the phase shift is computed by $p_n^\pm = \arg[A_n^\pm/|A_n^\pm|]$. For TM polarization, there holds $p_n^\pm = \arg[B_n^\pm/|B_n^\pm|]$. Finally, in the case of conical diffraction, the phase shift $p_n^{l,\pm}$ is that of the corresponding field component presented in the efficiency $e_n^{l,\pm}$ in accordance with the chosen output type (cf. Section 2.3). In particular, for output type "3.Com" (first type in Section 2.3) and incidence angle $\phi = 0$, the phase shifts of the classical case are computed. In the case of output type "TE/TM" (second type) and incidence angle $\phi = 0$, the phase shifts computed by the conical case are either equal to those of the classical or they differ by an angle of $\pm 180°$. Equality corresponds to a vector $s_n^\pm$ equal to the unit vector in the direction of the z coordinate and deviation by $\pm 180°$ to $s_n^\pm$ equal to the unit vector in the direction opposite to the z coordinate.

consists in finding a parameter set r_{opt} for which the value $f(r_{opt})$ is less or equal to $f(r)$ for all r in the set of the admissible parameter sets of the class.

The arithmetic expression for $f(r)$ looks a little bit complicated and is defined as

$$\begin{aligned}
f(r) &:= \sum_{j=1}^{j_\lambda}\sum_{k=1}^{k_\theta}\sum_{l=1}^{l_\phi}\sum_{m=1}^{m_t} S_{j,k,l,m}, \qquad (10.2)\\
S_{j,k,l,m} &:= \bar{w}^+e^+ + \bar{w}^-e^- + \bar{w}\,e + \sum_n \bar{w}_n^+e_n^+ + \sum_n \bar{w}_n^-e_n^- + \sum_n \bar{w}_n^{1,+}e_n^{1,+} + \\
&\sum_n \bar{w}_n^{1,-}e_n^{1,-} + \sum_n \bar{w}_n^{2,+}e_n^{2,+} + \sum_n \bar{w}_n^{2,-}e_n^{2,-} + \sum_n w_n^+\left[e_n^+ - c_n^+\right]^2 + \\
&\sum_n w_n^{1,+}\left[e_n^{1,+} - c_n^{1,+}\right]^2 + \sum_n w_n^{2,+}\left[e_n^{2,+} - c_n^{2,+}\right]^2 + \sum_n w_n^-\left[e_n^- - c_n^-\right]^2 + \\
&\sum_n w_n^{1,-}\left[e_n^{1,-} - c_n^{1,-}\right]^2 + \sum_n w_n^{2,-}\left[e_n^{2,-} - c_n^{2,-}\right]^2 + w^+\left[e^+ - c^+\right]^2 + \\
&w^-\left[e^- - c^-\right]^2 + w\left[e - c\right]^2 + \\
&\sum_n \tilde{w}_n^{1,+}\left[p_n^{1,+} - \tilde{c}_n^{1,+}\right]_*^2 + \sum_n \tilde{w}_n^{2,+}\left[p_n^{2,+} - \tilde{c}_n^{2,+}\right]_*^2 + \\
&\sum_n \tilde{w}_n^{1,-}\left[p_n^{1,-} - \tilde{c}_n^{1,-}\right]_*^2 + \sum_n \tilde{w}_n^{2,-}\left[p_n^{2,-} - \tilde{c}_n^{2,-}\right]_*^2, \\
\left[p_n^\pm - \tilde{c}_n^\pm\right]_*^2 &:= \left[\sin\left(\frac{p_n^\pm - \tilde{c}_n^\pm}{2}\right)\right]^2,
\end{aligned}$$

where the $\bar{w}^\pm$, $\bar{w}$, $\bar{w}_n^\pm$, $\bar{w}_n^{l,\pm}$, $w_n^\pm \geq 0$, $w_n^{l,\pm} \geq 0$, $w^\pm \geq 0$, $w \geq 0$ $\tilde{w}_n^\pm$, $\tilde{w}_n^{l,\pm}$, $c_n^\pm$, $c_n^{l,\pm}$, $c^\pm$, c, $\tilde{c}_n^\pm$, and $\tilde{c}_n^{l,\pm}$ are constants fixed by the user. Moreover, these constants can be chosen in dependence on λ_j, θ_k, ϕ_l, and t_m. Note that the phase shift is an angle between $-180°$ and $180°$. Since the angles $-180°$ and $180°$ correspond to the same phase shift, we have replaced the quadratic term $[p_n^+ - \tilde{c}_n^+]^2$ by its periodic variant $[p_n^+ - \tilde{c}_n^+]_*^2$. Choosing $\tilde{w}_n^\pm > 0$, i.e. including phase shift terms, can be dangerous. Namely, if the corresponding efficiency is zero, then the phase shift is not defined. Even for small efficiencies, a very fine FEM grid is needed to obtain acceptable approximate values for the phase shifts.

For example, if the user wants to design a beam splitter into four transmitted directions, then he would choose, e.g., the objective functional

$$f(r) = \left[e_{-1}^- - 25\right]^2 + \left[e_0^- - 25\right]^2 + \left[e_1^- - 25\right]^2 + \left[e_2^- - 25\right]^2. \qquad (10.3)$$

Choosing the functional $f(r) = e^+$, would result in some kind of anti-reflection grating. If the user is concerned with the synthesis problem (inverse problem), then the efficiencies are determined by measurements and the grating realizing these values is sought. Knowing that the sought grating is in a certain class of gratings described by the parameter sets r, the inverse problem is equivalent to the minimization of $f(r) = \sum_{n,\pm}[e_n^\pm - c_n^\pm]^2$ with $c_n^\pm$ chosen as the corresponding measured efficiencies.

If the type of polarization and the coordinate system for the incoming wave vector is "TE/TM", then the phase shifts are computed first for the TE polarization and then for the TM case. We denote this dependence by adding a (TE) and (TM), respectively. Instead of the terms

$$\sum_n \tilde{w}_n^{1,+}\left[p_n^{1,+}-\tilde{c}_n^{1,+}\right]_*^2+\sum_n \tilde{w}_n^{2,+}\left[p_n^{2,+}-\tilde{c}_n^{2,+}\right]_*^2+ \\ \sum_n \tilde{w}_n^{1,-}\left[p_n^{1,-}-\tilde{c}_n^{1,-}\right]_*^2+\sum_n \tilde{w}_n^{2,-}\left[p_n^{2,-}-\tilde{c}_n^{2,-}\right]_*^2 \tag{10.4}$$

in (10.2) one might wish to include

$$\sum_n \tilde{w}_n^{1,+}\left[p_n^{1,+}(\mathrm{TE})-p_n^{1,+}(\mathrm{TM})-\tilde{c}_n^{1,+}\right]_*^2+\sum_n \tilde{w}_n^{2,+}\left[p_n^{2,+}(\mathrm{TE})-p_n^{2,+}(\mathrm{TM})-\tilde{c}_n^{2,+}\right]_*^2+ \\ \sum_n \tilde{w}_n^{1,-}\left[p_n^{1,-}(\mathrm{TE})-p_n^{1,-}(\mathrm{TM})-\tilde{c}_n^{1,-}\right]_*^2+\sum_n \tilde{w}_n^{2,-}\left[p_n^{2,-}(\mathrm{TE})-p_n^{2,-}(\mathrm{TM})-\tilde{c}_n^{2,-}\right]_*^2. \tag{10.5}$$

To indicate that terms of this type are to be included instead of the corresponding terms depending on one polarization only, one easily adds a “TE/TM” before the input of the number of terms (cf. Section 10.1.3).

Similarly, if the type of polarization is “TE/TM”, if the illumination is classical (angle of incidence $\phi = 0$), and if the output data is presented in the “TE/TM” form (cf. Section 2.3), then the objective functional might contain terms with the phase difference between the classical TE and TM case. In other words, one might need terms like

$$\sum_n \tilde{w}_n^{1,+}\left[p_n^{1,+}(\mathrm{TE})-p_n^{2,+}(\mathrm{TM})-\tilde{c}_n^{1,+}\right]_*^2+\sum_n \tilde{w}_n^{1,-}\left[p_n^{1,-}(\mathrm{TE})-p_n^{2,-}(\mathrm{TM})-\tilde{c}_n^{1,-}\right]_*^2. \tag{10.6}$$

Such terms are indicated by adding a “CL:TE/TM” before the input of the number of terms (cf. Section 10.1.3).

10.1.2 Optimization via `OPTIMIZE`

In order to solve the above optimization problem, the user changes to the subdirectory `OPTIM`. Here he finds the executable `OPTIMIZE`, the input file “conical.Dat” with the control parameters of the generalized FEM (cf. the description of the similar file “generalized.Dat” in Section 6), as well as the example input file “example.dat” (cf. the enclosed data file in Section 12.8). At first, the user creates his own input file, e.g. “name.dat”, containing all the data of the grating geometry, of the illumination conditions, the initial values of the parameters[20] to be optimized, the data of the objective function, and that of the method

[20] If no good initial solution is known, then the user can try to find one by a deterministic search algorithm over a tensor product grid of parameter points. In other words, the initial solution of parameter values is the parameter point of the grid at which the objective functional takes its minimum over the finite grid. Instead of a list of initial values the user writes a line like “ no n_{lev} n_{nmb} 0” into the input file. Here n_{lev} is the discretization level of the FEM method at which the search is to be accomplished. The second number n_{nmb} is the maximal number of grid points in one dimension. This is attained at least in the direction of the longest side of the box domain defined by the upper and lower parameter bounds. If the user adds the input line “ no n_{lev} n_{nmb} 1”, then the minimum search over the grid points is improved by replacing the value of the objective functional at each grid point by the minimum of the linear Taylor approximation taken over a small neighbourhood of the grid point.

of optimization. The best way to do this is to copy “example.dat” to “name.dat” and to change the data to meet the requirements of the user. Numerous comments contained in this file will make it easy to get the right input (for some more details cf. the subsequent Section 10.1.3). Then the user enters the command

```
OPTIMIZE name
```

and the optimization starts. All methods of optimization are iterative. The number of the actual iteration step and the value of the objective function at the actual iterative solution are printed on the screen. If the last iteration step is finished, there appears the minimal value of the objective functional and the corresponding set of optimal parameters on the screen. Finally, in the directory `RESULTS` a file “name2.res” (cf. the enclosed result file in Section 12.9) is created which contains the input data of the optimization problem, some data of the iteration process, and the optimal solution. The name “name2” is to be fixed by the user in the input file “name.dat”.

Clearly, the optimization methods for (10.1) are based on the same finite element discretization which is implemented for the simulation of diffraction (cf. Sections 2 and 5-7). Even the gradients of the objective functions can be computed using the same finite element matrices. If the efficiencies and phase shifts are determined with a discretization error ε_{dis}, then the error of the objective functional evaluation is about ε_{dis} and the error of the numerical approximation to the minimal value $f(r_{opt})$ is expected to be about ε_{dis}. Unfortunately, the numerical approximation of the optimal solution r_{opt} is expected to have an error about $\sqrt{\varepsilon_{dis}}$, only.

Alternatively to the simple call of `OPTIMIZE`, the user may enter one of the following commands: If `OPTIMIZE` is called with the flag `-f`, then the optimization is performed like in the case without flag, but further data on the several iteration steps will be printed on the screen. With the flag `-s`, the optimization is performed like in the case without flag and, afterwards, a plot of the resulting optimal grating is shown on the screen. The call of `OPTIMIZE` with one of the flags `-i`, `-g`, and `-p` does not invoke any optimization method. Such a call is designed to prepare the optimization. The flag `-i` results in an input check which includes a plot of the initial grating (like those in Figures 29 and 31) and the computation of the corresponding value of the objective functional. A call with `-g` checks the local error of the gradient computation. An approximation for the absolute and relative error of the gradient at the initial solution is determined. Since the true value of the gradient is unknown, the true gradient is replaced by the approximate gradient computed on a refined FEM mesh.

If `OPTIMIZE` is called with the flag `-f`, then the value of the objective function and the parameters of the actual iterate are shown on the screen. After a few iteration steps the differences in the objective function and the parameters of consecutive iterates might be very small such that the user wants to stop the iteration and to switch to the next higher level determined by the incremental input for the level. This can be accomplished with the `kill` command. The user opens another window and enters

```
ps -al | grep optim
```

in order to find out the process number PID of the process `optimize`, `optimize_m`, resp. `optimize_l`. Then he enters

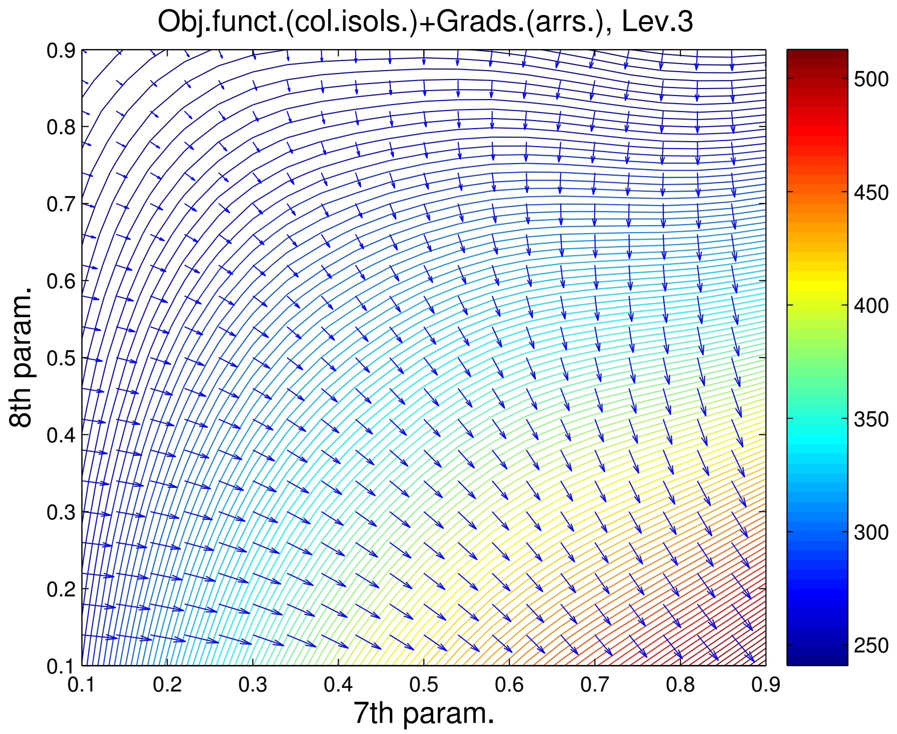


Figure 27: `Matlab` plot of an objective functional and its gradient field.

```
kill -10 PID
```

with PID replaced by the PID number of the process. The flag `-10` of the `kill` command raises the signal SIGUSR1 (cf. e.g. the manual page signal(7) of the linux system). If the optimization catches this signal, then the iteration is interrupted.

For the gradient based numerical optimization schemes (cf. Sections 10.3.2-10.3.4), the successful computation of the gradient is essential. Therefore, a visualization should help to control the gradient approximation. In case of inaccurate gradients, the discretization level must be refined. The call of `OPTIMIZE` together with the flag `-p` results in a `Matlab` code file "MLplot.m" for a plot of the objective function and the gradient field over a two-dimensional parameter set (cf. Figure 27). The code can be started calling "MLplot" from `Matlab`. In the case of higher dimensional optimization problems, the user can choose any pair of two parameters and fix the others in the input file "name.dat" by setting upper and lower bounds to equal values. This way any two-dimensional section of the graph of the objective function and the gradient field can be displayed.

10.1.3 The input file for the optimization

An input file "name.dat" (cf. the enclosed data file in Section 12.8) in the subdirectory `OPTIM` is needed. To get this, change the directory to `OPTIM`, copy one of the existing files with tag ".dat", e.g. the file "example.dat" and call it "name.dat".

```
cd DIPOG-2.1/CLASSICAL
```

```
cp example.dat name2.dat
```

Change "name.dat" in the editor according to your requirements. You will find the necessary information as comments in the file "name.dat". Indeed, each line beginning with "#" is a comment. Comment lines can be added and deleted without any trouble.

The first input of "name.dat" is the name "name2" of the result file "name2.res" without the tag ".res" which will be written into the subdirectory `RESULTS`. This is done by the lines:

```
# Name of output file.
   name3
```

These lines are followed by several inputs concerning the grating data which are similar to those in the input files for the conical diffraction (cf. the file of Section 12.3 and compare Section 6). Of course, the geometry description is different since now the geometry of the grating is to be optimized. The geometry is described e.g. by the lines:

```
# Number nd_geom_param of real parameters:
  6
# Lower bounds dl_geom_param:
  -0.5
  -0.5
  1.
  0.
  1.5
  0.
# Upper bounds du_geom_param:
  0.5
  0.5
  1.
  0.
  1.5
  0.
# Number ni_geom_param of integer parameters:
  3
# Integer parameters i_geom_param:
  1
  2
  2
# Number ns_geom_param of name parameters:
  0
# Parameter names s_geom_param:
########################################
# Parameters d_geom_param of initial grating:
  0.07
  -0.04
  1.
```

```
    0.
    1.5
    0.
```

Thus the parameter set of the grating geometry contains nd_geom_param=6 real parameters, ni_geom_param=3 integers, and ns_geom_param=0 names (character strings). The integer and name parameters are given in the lines following their number. The first integer parameter i_geom_param[1] is the index of the grating class. The meaning of all the other parameters depends on the class and will be explained in Section 10.2. In general, the real parameters d_geom_param[i] are the parameters r_i subject to the optimization. The upper and lower bounds u_i and l_i are the numbers du_geom_param[i] and dl_geom_param[i], respectively. Clearly, the choice du_geom_param[i]=dl_geom_param[i] would fix the parameter and only those d_geom_param[i] with du_geom_param[i] greater than the value dl_geom_param[i] are optimized. Note that the geometry description includes geometry parameters like heights, widths, and lengths as well as material parameters like the refractive indices.

In the following input lines of "name.dat" the level of discretization is given. Level equal to l means that the FEM grid for the computation of the objective function and its gradient is created with a maximal stepsize of $h_0 2^{-l}$, where h_0 is the coarsest stepsize. For example, the level is set to 3 by

```
# Number of levels:
    3
```

If the computation of the objective functional is time consuming, then it might be better to perform a few iteration on a coarser FEM level first, and to utilize the coarse level optimal solution as the initial solution on the fine level. Therefore, an incremental input is possible. For instance,

```
# Number of levels:
    I 1 5 2
```

indicates that the level takes all values $1 + j * 2$ with $j = 0, \dots$ such that $1 + j * 2 \leq 5$. Hence, using the initial values, the optimization starts with an iteration on level 1. The optimal solution of level 1 is passed to the initial solution of the level 3 iteration. After the level 3 iteration is finished, the level 3 solution will be the initial solution of level 5. Hopefully, this is a better initial solution than that given by the user in "name.dat", and, after a smaller number of iterations, the optimal solution of level 5 is reached.

After the input of the level, the parameters $\bar{w}^{\pm}$, $\bar{w}$, $\bar{w}_n^{\pm}$, $\bar{w}_n^{l,\pm}$, $w_n^{\pm}$, $w_n^{l,\pm}$, $w^{\pm}$, w $\tilde{w}_n^{\pm}$, $\tilde{w}_n^{l,\pm}$, $c_n^{\pm}$, $c_n^{l,\pm}$, $c^{\pm}$, c, $\tilde{c}_n^{\pm}$, and $\tilde{c}_n^{l,\pm}$ of the objective functional f in (10.2) are given. For instance, the lines

```
#QUADRATIC TERMS, TRANSMITTED EFFICIENCY
# n_qua_tr:
    2
# w_qua_tr:
    2.
    3.
```

```
# o_qua_tr:
  -1
  0
# c_qua_tr:
  10.
  20.
```

indicate that the number of quadratic efficiency terms with non-zero weight factor is two, i.e. there are exactly two terms in the sum $\sum_n w_n^-[e_n^- - c_n^-]^2$ of (10.2). The corresponding orders n of the transmitted plane wave mode are -1 and 0, the weight factors $w_{-1}^- = 2$ and $w_0^- = 3$, and the prescribed efficiency values $c_{-1}^- = 10$ and $c_0^- = 20$. In other words, the term $\sum_n w_n^-[e_n^- - c_n^-]^2$ in (10.2) turns into $2[e_{-1}^- - 10]^2 + 3[e_0^- - 20]^2$. The lines

```
#QUADRATIC TERMS, FIRST (TE or S) TRANSMITTED EFFICIENCY
# n_qua_1_tr:
  1
# w_qua_1_tr:
  2.
# o_qua_1_tr:
  -1
# c_qua_1_tr:
  10.
```

indicate that the number of first type[21] quadratic efficiency terms with non-zero weight factor is one, and the sum $\sum_n w_n^{1,-}[e_n^{1,-} - c_n^{1,-}]^2$ turns into $2[e_{-1}^{1,-} - 10]^2$. Moreover, the values $c_n^\pm$, $c_n^{l,\pm}$, $\tilde{c}_n^\pm$, $\tilde{c}_n^{l,\pm}$, c, and $c^\pm$ may depend on the wave lengths λ_j, on the angles of incidence θ_k and ϕ_l, and on the polarization type t_m. If e.g. the wave length runs through the values $\lambda_j, j = 1, 2, 3 = j_\lambda$, then the $j_\lambda \times$ n_qua_tr = 6 different values can be fixed, e.g., by

```
#QUADRATIC TERMS, TRANSMITTED EFFICIENCY
# n_qua_tr:
  2
# w_qua_tr:
  WAL
  2.
  3.
  2.
  3.
  2.
  3.
# o_qua_tr:
  -1
  0
```

[21]First type efficiency means efficiency of the TE part of the mode if the type of the output is set to TE/TM (second variant of output) and the S-part of the mode if the type of the output is set to *Jones* (third variant of output in Section 2.3).

```
# c_qua_tr:
    WAL
    10.
    15.
    20.
    25.
    30.
    35.
```

In this case, the expression $\sum_j \sum_n w^-_{f,n}[e^-_n(\lambda_j) - c^-_n(\lambda_j)]^2$ turns into:

$$2\left[e^-_{-1}(\lambda_1) - 10\right]^2 + 3\left[e^-_0(\lambda_1) - 15\right]^2 + 2\left[e^-_{-1}(\lambda_2) - 20\right]^2 + 3\left[e^-_0(\lambda_2) - 25\right]^2 + \\ 2\left[e^-_{-1}(\lambda_3) - 30\right]^2 + 3\left[e^-_0(\lambda_3) - 35\right]^2 .$$

Note that in our example, though only the values c^-_n depend on the wave length, the input of the corresponding weight numbers is done in the same mode. In general, the weights $w^{\cdots}_{\cdots}$ and the corresponding prescribed values $c^{\cdots}_{\cdots}$ must always be given in the same mode. In other words, if one of the two is constant and the other depends on an entity, then the constant values must be repeated to get the same input form.

Replacing, WAL by ATH, APH or POL and the following $j_\lambda \times$ n_qua_tr values for c^-_n and w^-_n by $k_\theta \times$ n_qua_tr, $l_\phi \times$ n_qua_tr, and $m_t \times$ n_qua_tr values, respectively, the user can define the values c^-_n and w^-_n depending on the angles θ and ϕ, and on the polarization type, respectively. If the user replaces WAL by W+T, W+P, T+P, WAL+POL, ATH+POL, and APH+POL and the following $j_\lambda \times$ n_qua_tr values by $j_\lambda \times k_\theta \times$ n_qua_tr, $j_\lambda \times l_\phi \times$ n_qua_tr, $k_\theta \times l_\phi \times$ n_qua_tr, $j_\lambda \times m_t \times$ n_qua_tr, $k_\theta \times m_t \times$ n_qua_tr, and $l_\phi \times m_t \times$ n_qua_tr values, respectively, then the user can fix the dependence on wave length plus θ, on wave length plus ϕ, on θ plus ϕ, on wave length plus polarization type, on θ plus polarization type, and on ϕ plus polarization type, respectively. Choosing WTP, W+T+POL, W+P+POL, and T+P+POL for WAL as well as $j_\lambda \times k_\theta \times l_\phi \times$ n_qua_tr, $j_\lambda \times k_\theta \times m_t \times$ n_qua_tr, $j_\lambda \times l_\phi \times m_t \times$ n_qua_tr, and $k_\theta \times l_\phi \times m_t \times$ n_qua_tr, values, respectively, the dependence on wave length plus θ plus ϕ, on wave length plus θ plus polarization type, on wave length plus ϕ plus polarization type, and on θ plus ϕ plus polarization type, respectively, is managed. Replacing WAL by WTP+POL and adding $j_\lambda \times k_\theta \times l_\phi \times m_t \times$ n_qua_tr input numbers, the dependence on wave length plus θ plus ϕ plus polarization type is obtained[22].

As mentioned in Section 10.1.1, adding a "TE/TM" before the input of n_phs_j_re resp. n_phs_j_tr the terms (10.4) can be replaced by (10.5). E.g., for the second type reflected phase shifts the corresponding input can look like:

```
# n_phs_2_re:
    TE/TM
    2
```

In the case of the terms (10.5) an input of the weights w_phs_j_re resp. w_phs_j_tr and of

[22]The values for c^-_n and w^-_n, respectively, are given in a nested loop over the wave length, angle θ, angle ϕ, polarization type, and the index of the mode. The innermost loop is over the n_qua_tr modes. The next is over the m_t polarization types, the next over the l_ϕ angles ϕ, and the next over the k_θ angles θ. Finally, the outermost loop is over the j_λ values of the wave length.

the prescribed values c_phs_j_re resp. c_phs_j_tr beginning with a sequence containing POL is not possible since the phase shifts for different polarization are included into one term, only. Finally, suppose that input and output are of type "TE/TM" and that the angle of incidence ϕ is zero (cf. Section 2.3). Then, adding a "CL:TE/TM" before the input of n_phs_1_re resp. n_phs_1_tr, the terms (10.4) are replaced by (10.6). This looks like

```
# n_phs_1_tr:
  CL:TE/TM
  2
```

In general, the objective functional may depend on 999 different values of efficiencies/phase shifts/energies. If more than 999 efficiencies/phase shifts/energies are needed the environment variable `NMB_OF_DATA` must be set to a number larger than the number of required efficiencies/phase shifts/energies.

Finally, the file "name.dat" has to fix the scheme of numerical optimization together with its control parameters. Generally, this looks like

```
# Maximal number of iterations:
  10000
# Indicator for optimization method:
  1
# ni_opt:
  1
# i_opt:
  3
# nd_opt:
  5
# d_opt:
  1.
  0.001
  1e-3
  1e-2
  1e-2
# Scaling parameters d_geom_scal:
  1.
  1.
  1.
```

Here the indicator 1 is the index of the numerical method. The number and the meaning of the other parameters depend on the indicator value. They will be explained in Section 10.3. In the example from above the numerical method of index 1 requires one integer parameter i_opt[1] and five real control parameters d_opt[1], d_opt[2], d_opt[3], d_opt[4], and d_opt[5]. Beside the control parameters, the successful run of the local optimization routines depends on the scaling of the parameters d_geom_param[j] by the scaling factors d_geom_scal[j], $j = 1, \dots$,nd_geom_param (cf. Section 10.3.1).

If an optimization over more than one level is performed, then the maximal number of iterations can be chosen in dependence on the level. For instance, if the input of the level is the incremental “I 1 5 2”, then there are three different levels 1, 3, and 5. The input

```
# Maximal number of iterations:
  LEV
  10
  20
  10000
```

would restrict the number of level 1 iteration to 10, that of level 3 to 20, and that of level 5 to 10000.

10.1.4 The locality of the solution. A warning

The parameter set $r_{opt} = \{r_i : i = 1, \dots, N\}$ for which $f(r_{opt}) \le f(r)$ for all r in the fixed class of admissible parameters is called the global optimal solution. Since the objective functional is continuous and since the class of parameters is compact, the existence of r_{opt} is guaranteed. However, the optimal solution is, in the general case, not unique. Moreover, the topography of the graph of the objective function is often quite complex. Usually, there exist a lot of local minima. Note that a set of parameters r_{loc} is called local minimum if, for all r close to r_{loc}, the value $f(r_{loc})$ is less or equal to $f(r)$. Sometimes, there exists even a submanifold in the class of admissible parameter sets consisting of local minima.

Unfortunately, the final result of the numerical optimization schemes is often a local minimum instead of global optimum. Indeed, the gradient based methods (cf. Sections 10.3.2-10.3.4 and 10.3.6) are so called local methods which are designed to determine a local minimum. To find a global minimum, the user should start the local methods from several sets if initial parameters and to take the local optimal solution with the minimal value of the objective functional as a global solution. Of course, there is no warranty that the true global minimum has been found. On the other hand, frequently, the local minimizer corresponds to a value of the objective function quite close to the minimum. Such a local minimum may be just as good as the global optimal solution.

Clearly, one can utilize optimization methods designed to find global minimizers. Usually, these global optimizers are much slower than local methods. Other global optimizers use more information on the optimization problem which are not natural in our applications. The only global algorithm in the package `DIPOG`-2.1 is the simulated annealing (cf. Section 10.3.5) which is a stochastic method. In other words, if the parameters are chosen adequately, then the solution of simulating annealing is a global solution with probability one. Choosing the right parameters, however, is not easy. Either the number of necessary iterations might be large and the computing time might be not acceptable or the algorithm might stuck in a local minimizer. So there is no warranty even with the global simulated annealing.

Finally, let us mention that the user is free to combine global and local methods. First he should apply a certain number of simulated annealing steps. Using the final solution of simulated annealing as the initial solution, he should apply a local method to end up with an improved final solution.

10.1.5 Maximum likelihood estimator

Suppose only efficiency data is measured and suppose that the measured values are sample values of random variables. More precisely, suppose that the measured $c_n^{i,\pm}$ is normally distributed with expectation $e_n^{i,\pm}(r_{opt})$ and variance

$$[\sigma_n^{i,\pm}]^2 = a^2[e_n^{i,\pm}(r_{opt})]^2 + b^2,$$

where b is the back ground noise level and where a is a yet unknown constant factor. If the measurements are independent, then the joint distribution of the measurement data takes the form

$$\varrho\Big(\{c_n^{i,\pm}\}\Big) = \varrho\Big(a,r;\{c_n^{i,\pm}\}\Big) = \prod_{i,\pm,n} \frac{1}{\sqrt{2\pi}\sqrt{a^2[e_n^{i,\pm}(r_{opt})]^2+b^2}}\, e^{-\frac{(c_n^{i,\pm}-e_n^{i,\pm}(r_{opt}))^2}{2[a^2[e_n^{i,\pm}(r_{opt})]^2+b^2]}}.$$

Hence, given a fixed measurement data $\{c_n^{i,\pm}\}$, the maximum likelihood estimator determines those model parameters r and a for which the density $\varrho(a,r;\{c_n^{i,\pm}\})$ is maximal. In other words, the maximum likelihood approach determines r and a minimizing the functional

$$f(r,a) := \sum_{i,\pm,n} \left\{ \log\left(a^2[e_n^{i,\pm}(r)]^2+b^2\right) + \frac{\left(c_n^{i,\pm}-e_n^{i,\pm}(r)\right)^2}{a^2[e_n^{i,\pm}(r)]^2+b^2} \right\}$$

over the domain defined by $0 \le a$ and $l_i \le r_i \le u_i$.

The optimization can be switched to this maximum likelihood method by changing the input data file "name.dat" as follows. Firstly, all weights for energies and all numbers for the involved linear efficiency resp. quadratic phase-shift terms must be set to zero. Secondly, the weight inputs must be like

```
# QUADRATIC TERMS, FIRST (TE or S) REFLECTED EFFICIENCY
  Max_Like  a  b
  0.
  1.
  ...
```

Here the symbol a stands for the initial guess of the variance factor a and b for the fixed back ground noise level b. The "weight" numbers following after the first line must be listed like the weights of quadratic efficiency terms. Their values, however, must be zero or one. A one indicates that the term with the corresponding efficiency is included into the summation for the objective functional f and a zero that it is not.

10.2 Classes of gratings which can be optimized

10.2.1 General parameters

For the description of the geometry nd_geom_param real, ni_geom_param integer, and ns_geom_param name parameters (character string with less than 250 characters) are needed.

These are stored in the vectors:

d_geom_param[j], $j = 1, \dots$, nd_geom_param
i_geom_param[j], $j = 1, \dots$, ni_geom_param
s_geom_param[j], $j = 1, \dots$, ns_geom_param

The integer and name parameters are fixed by the user. The vector of real parameters will be optimized. Therefore, initial values of these d_geom_param[j], $j = 1, \dots$, nd_geom_param are to be fixed by the user. Moreover, the real parameters are restricted to intervals. The upper bounds du_geom_param[j] and lower bounds dl_geom_param[j] of the intervals are given by the user. The user can even fix a real parameter to a constant value setting the corresponding upper bound equal to the lower.

Some of the real parameters are the real or imaginary part of a refractive index (cf. the following subsections of Section 10.2). Suppose now that this should be fixed to a constant value, i.e. the real parameter is not included into the set of optimization parameters. Similarly as for the simulation by `DIPOG`-2.1 (cf. Section 4), such parameters can be chosen from predefined lists. For instance, if the refractive index in the simulation would be determined by the name “AlAs”, then the parameter input d_geom_param[j] of the real part of the refractive index can now be given as the string “Re AlAs”. Clearly, the parameter input d_geom_param[k] of the corresponding imaginary part of the refractive index is to be given as “Im AlAs”. In any case, if a real resp. imaginary part of a refractive index is determined by such a string input, then the corresponding imaginary resp. real part must be defined the same way. Moreover, the corresponding bounds dl_geom_param[j], du_geom_param[j], dl_geom_param[k], and du_geom_param[k] must be determined with the same input strings. Note that, for more than one wavelengths involved in the computation of the objective functional, `DIPOG`-2.1 cannot optimize the refractive indices depending on the wavelengths. Therefore, for multiple wavelengths, all real parameters must be fixed to a “constant” value independent of the optimization process but, of course, depending on the wavelength. In other words, for multiple wavelengths, the input by strings is mandatory.

In some applications with a large number of geometrical parameters, only a small number of parameters can be chosen freely and the other depend on these free parameter through explicit formulas. If the jth real parameter d_geom_param[j] is a function of the parameters d_geom_param[k_l], $l = 1, \dots , L$, then the input of the bounds dl_geom_param[j] and du_geom_param[j] and the initial value d_geom_param[j] is to be replaced by adding the dependency function. More precisely, instead of the numbers for du_geom_param[j] and d_geom_param[j] the word “Dep” indicates that the corresponding parameter depends on other parameters. The number for dl_geom_param[j] must be replaced by the string “Dep:” followed by the dependency function. This function is to be written as a c-code, where the argument d_geom_param[k_l] is denoted by “pk_l”. For instance, the dependency

$$\text{d\_geom\_param}[3] = \frac{\text{d\_geom\_param}[7] \cdot \text{d\_geom\_param}[8]}{\text{d\_geom\_param}[2]}$$

together with the values d_geom_param[1]=0.5 bounded to the prescribed interval [0.1,0.9] and d_geom_param[2]=0.3 from the interval [0.2,0.4] is indicated by the input lines

```
# Lower bounds dl_geom_param:
  0.1
  0.2
  Dep: p7*p8/p2
  ...
# Upper bounds du_geom_param:
  0.9
  0.4
  Dep
  ...
# Number ni_geom_param of integer parameters:
  ...
# Integer parameters i_geom_param:
  ...
# Number ns_geom_param of name parameters:
  0
# Parameter names s_geom_param:
##########################################
# Parameters d_geom_param of initial grating:
  0.5
  0.3
  Dep
  ...
```

To avoid contradictions, the arguments d_geom_param[k_l] of a dependency function must be free parameters.

If possible one should try to present rational dependency in the standard form

Dep: f_1+(f_2)/(f_3)

where each f_i stands for an expression including the variables pj, constants, and the operations "+", "-", and "$*$" but no blanks and no brackets. The code realizes that the dependency is rational and generates code for the derivatives. If the dependency is not rational or not in the standard form, then the derivatives are approximated by difference formulas.

The number of integer parameters is larger or equal to two. The first i_geom_param[1] is an index between one and six indicating that the grating belongs to one of the six grating classes.

- i_geom_param[1]=1: Profile grating determined by a polygonal profile function
- i_geom_param[1]=2: Profile grating determined by general polygonal profile curve I
- i_geom_param[1]=3: Profile grating determined by general polygonal profile curve II
- i_geom_param[1]=4: Stack of trapezoids
- i_geom_param[1]=5: General grating with polygonal interface to be optimized
- i_geom_param[1]=6: Bridge composed of trapezoids under light in the EUV range

The second number i_geom_param[2] is the number of different materials contained in the grating including the cover and substrate material.

In general, the code will generate geometry input files of the form described in Section 3. The upper bound for the mesh size is fixed by the input lines

```
# Upper bound for mesh size:
  0.5
```

contained in the generated input files. However, this automatically chosen value 0.5 can be changed by the user setting the environment variable `BND_MESH_SIZE` to the desired value before calling the optimization routines.

10.2.2 Profile grating determined by a polygonal profile function

The first class (i_geom_param[1]=1) is a profile grating without coatings defined by a profile function which is the cross section of the interface between cover and substrate material. Thus the number of materials is i_geom_param[2]=2. The profile is supposed to be the graph of a piecewise linear function. The class has no name parameters (ns_geom_param=0) and ni_geom_param=3 integer parameters. The integer value i_geom_param[3] is the number N of knots of the profile curve in the interior of the period, i.e. with x-coordinate strictly between zero and period d. Hence, the grating profile is the polygonal curve connecting the points

$$(0,0),\ (h,y_1),\ (2h,y_2),\ \dots,\ (\mathrm{N}h,y_\mathrm{N}),\ (d,0)$$

where $h = d/(\mathrm{N}+1)$. The number of real input parameters nd_geom_param is equal to i_geom_param[3]+4, and the first i_geom_param[3] parameters define the profile curve by y_j =d_geom_param[j], $j = 1, \dots$,i_geom_param[3]. The last four parameters describe the materials. More precisely, the refractive indices of the cover and substrate materials are:

$$\text{d\_geom\_param[k+1]+i*d\_geom\_param}[k+2],$$
$$\text{d\_geom\_param[k+3]+i*d\_geom\_param}[k+4],$$

where k=i_geom_param[3]. Clearly, the real parts of these indices must be positive and the imaginary parts non-negative. The imaginary part of the cover material must vanish. Moreover, the refractive indices must be fixed, i.e.

$$\text{du\_geom\_param}[k]=\text{dl\_geom\_param}[k],$$
$$k=\text{i\_geom\_param[3]+1}, \dots ,\text{i\_geom\_param[3]+4}.$$

10.2.3 Profile grating determined by general polygonal profile curve I

The second class (i_geom_param[1]=2) is a profile grating without coatings defined by a profile curve which is the cross section of the interface between cover and substrate material. Thus the number of materials is i_geom_param[2]=2. The profile is supposed to be a polygonal curve. The class has no name parameters (ns_geom_param=0) and ni_geom_param=3 integer parameters. The value i_geom_param[3] is the number N of knots of the profile curve in the interior of the period, i.e. with x-coordinate strictly between zero and period d. In other words, the grating profile is the polygonal curve connecting the points

$$(0,0),\ (x_1,y_1),\ (x_2,y_2),\ \dots,\ (x_N,y_\mathrm{N}),\ (d,0)$$

with arbitrary real values x_j and y_j such that $0 < x_j < d$ and such that the non-adjacent polygonal sides do not intersect each other (no self-intersection). The number of real parameters is nd_geom_param=2i_geom_param[3]+4 and the first 2i_geom_param[3] parameters define the profile curve by x_j =d_geom_param[2j-1], y_j=d_geom_param[2j], $j = 1, \ldots$,i_geom_param[3]. The last four parameters describe the materials. More precisely, the refractive indices of the cover and substrate materials are:

d_geom_param[k+1]+i*d_geom_param[$k+2$],
d_geom_param[k+3]+i*d_geom_param[$k+4$],

where k=2i_geom_param[3]. Clearly, the real parts of these indices must be positive and the imaginary parts non-negative. The imaginary part of the cover material must vanish. Moreover, the refractive indices must be fixed, i.e.

du_geom_param[k]=dl_geom_param[k],
k=2i_geom_param[3]+1, ... ,2i_geom_param[3]+4.

Given an arbitrary parameter set, the self-intersection of the polygonal curve is checked in the program. For simplicity, however, the corresponding restriction functionals are not included into the choice of search directions for the optimization algorithms (cf. Section 10.3). Hence, using this class and local optimization methods, only local minima away from the boundary of the domain of admissibility can be detected. The next class provides a proper treatment of the restriction functionals corresponding to self-intersection.

10.2.4 Profile grating determined by general polygonal profile curve II

The third class (i_geom_param[1]=3) is a profile grating without coatings defined by a profile curve which is the cross section of the interface between cover and substrate material. Thus the number of materials is i_geom_param[2]=2. The profile is supposed to be a polygonal curve. The class has no name parameters (ns_geom_param=0) and ni_geom_param=3 integer parameters. The value i_geom_param[3] is the number N of knots of the profile curve in the interior of the period, i.e. with x-coordinate strictly between zero and period d. In other words, the grating profile is the polygonal curve connecting the points

$$(0,0),\ (x_1,y_1),\ (x_2,y_2),\ \ldots,\ (x_N,y_\mathrm{N}),\ (d,0)$$

with arbitrary real values x_j and y_j such that $0 < x_j < d$ and such that some additional restrictions are satisfied. These additional restrictions guarantee that non-adjacent polygonal sides do not intersect each other. The number of real parameters nd_geom_param is equal to 2i_geom_param[3]+5 and the first 2i_geom_param[3] parameters define the profile curve by x_j =d_geom_param[2j-1], y_j=d_geom_param[2j], $j = 1, \ldots$,i_geom_param[3]. The next four parameters describe the materials. More precisely, the refractive indices of the cover and substrate materials are:

d_geom_param[k+1]+i*d_geom_param[$k+2$],
d_geom_param[k+3]+i*d_geom_param[$k+4$],

where k=2i_geom_param[3]. Clearly, the real parts of these indices must be positive and the imaginary parts non-negative. The imaginary part of the cover material must vanish. The last parameter d_geom_param[k] with the index k=2i_geom_param[3]+5 is a threshold parameter for the additional restrictions. This threshold and the refractive indices must be fixed, i.e.

du_geom_param[k]=dl_geom_param[k],
k=2i_geom_param[3]+1, ...,2i_geom_param[3]+5.

In order to introduce the additional restrictions, the corner points are denoted by $P_0 := (0,0)$, $P_j := (x_j, y_j)$, $j = 1, \ldots, \mathrm{N}$, and $P_{\mathrm{N}+1} := (d, 0)$. Excluding self-intersection means to guarantee:

- distance between neighbour corner points is positive
- each corner point is outside of an ellipsoidal neighbourhood of each polygonal side not containing the corner
- no intersection of non-adjacent sides of polygonal curve

Using the threshold parameter ε :=d_geom_param[k], the above restrictions can be reformulated as:

$$\begin{aligned} |P_i - P_{i+1}| &\geq \varepsilon d, \ i = 0, \ldots, \mathrm{N} \qquad (10.7) \\ |P_j - P_m| + |P_j - P_{m-1}| - |P_m - P_{m-1}| &\geq \varepsilon \, |P_m - P_{m-1}| \, , \ m = 1, \ldots, \mathrm{N}+1, \\ & \quad\; j = 0, \ldots, \mathrm{N}+1, \ j \neq m-1, m \\ \overline{P_{m-1}P_m} \cap \overline{P_{i-1}P_i} &= \emptyset, \ m, i = 1, \ldots, \mathrm{N}+1, \ |m-i| > 1. \end{aligned}$$

Note that the first two types of restrictions are included in the choice of the new search direction for the conjugate gradient algorithm (cf. Section 10.3.2). The last is automatically fulfilled for a new iterative solution if this is close to the last iterate.

10.2.5 Stack of trapezoids

The fourth class (i_geom_param[1]=4) is a flat grating with a stack of trapezoids put on the flat interface (cf. Figure 28 and the example in Figure 29). The trapezoids are located with their parallel sides in the direction of the x-axis, and one trapezoid is placed over the other such that the upper resp. lower sides of the two adjacent trapezoids coincide. All these trapezoids do not exceed the period $\{(x, y) : \ 0 \leq x \leq d\}$. If necessary, additional coating layers beneath the grating structure are allowed.[23]

The class has no name parameters (ns_geom_param=0) and ni_geom_param=2 integer parameters. The value i_geom_param[2] is the number of different materials which is equal to the number N of trapezoids plus two (two for substrate and cover material). The number of real parameters nd_geom_param is equal to 5·i_geom_param[2]-4=$5N + 6$. These parameters include five reals for each trapezoid, two reals for the location of the stack of trapezoids, and four reals for the refractive indices of the substrate and cover material. In particular (cf. Figure 28), if k is a positive integer less or equal to N, then the parameter d_geom_param[5k-4]=$h_k > 0$ is the height of the kth trapezoid measured in μm. The parameter d_geom_param[5k-3]=$b_k/d \in (0, 1]$ is the ratio of the x coordinate of the right upper corner of the kth trapezoid over the period d of the grating. The real d_geom_param[5k-2]=

[23] Unfortunately, the optimization is interrupted if the width of the first coating layer adjacent to the stack of trapezoids is less than the width of the additional strip automatically added to the Fem domain (cf. Section 3.2). The last width is the sum of two numbers. The first number is the minimum of the height of the lowest trapezoid and of half of the width of the first coating layer. The second number is the period multiplied by the minimum of the meshsize (0.5 to the power of the refinement level) and 0.05.

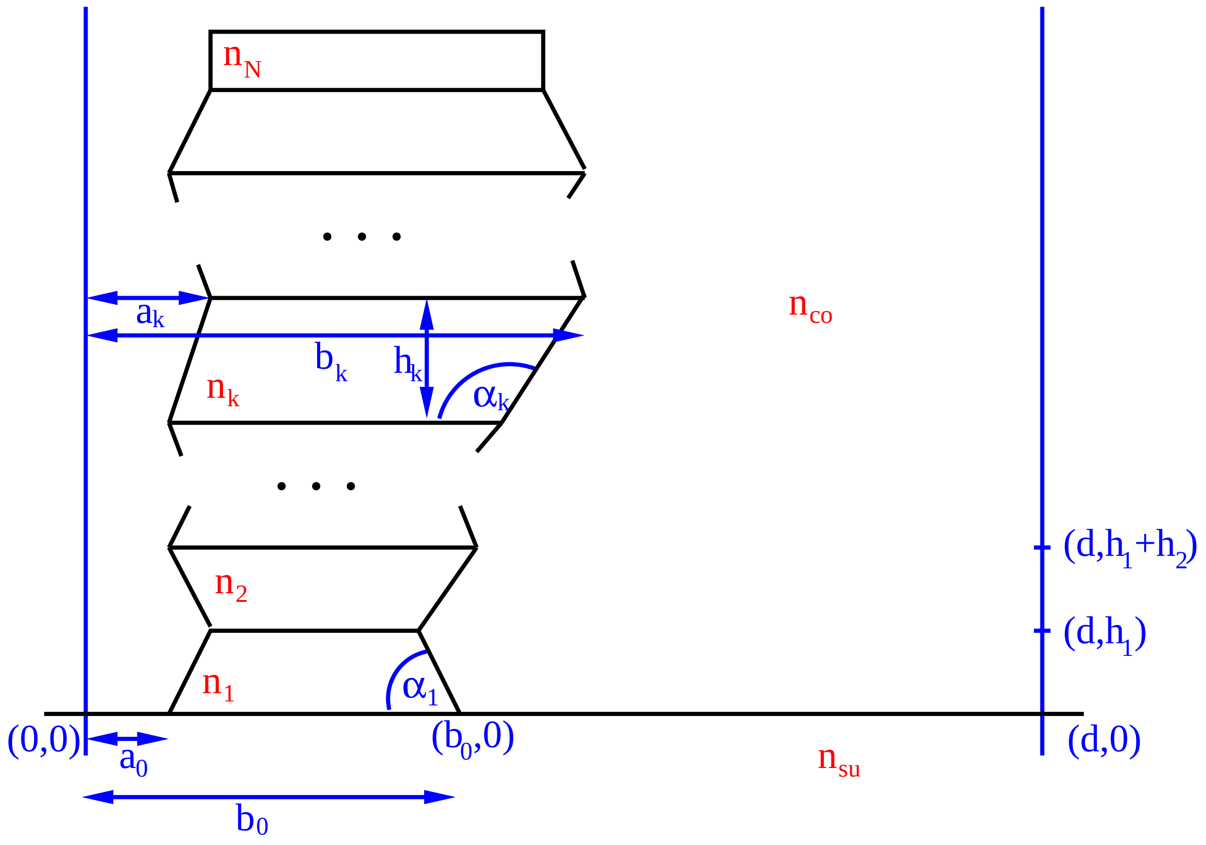


Figure 28: Stack of trapezoids.

$a_k/b_k \in [0,1)$ is the ratio of the x coordinate of the left upper corner of the kth trapezoid over the x coordinate of the right upper corner. The last parameters of the kth trapezoid form the refractive index n_k of the material, i.e. the refractive index is

n_k=d_geom_param[5k-1]+ i*d_geom_param[5k].

The next parameter d_geom_param[5N+1] is the ratio $b_0/d \in (0,1]$ of the x coordinate of the right lower corner of the first trapezoid over the period d of the grating. The next real parameter d_geom_param[5N+2]=$a_0/b_0 \in [0,1)$ is the ratio of the x coordinate of the left lower corner of the first trapezoid over the x coordinate of the right lower corner. Finally, the refractive indices of the cover and substrate material are given as

n_{co}=d_geom_param[5N+3]+i*d_geom_param[5N+4],
n_{su}=d_geom_param[5N+5]+ i*d_geom_param[5N+6].

Setting upper bound equal to lower bound, the user must fix these two refractive indices, i.e., the equality

du_geom_param[j]=dl_geom_param[j]

must be satisfied for $j = 5N+3, \ldots, 5N+6$. Clearly, the real parts of all refractive indices must be positive and the imaginary parts non-negative. The imaginary part of the cover material must vanish.

Note that the presented choice of real parameters has been adapted to the optimization problem in order to avoid additional restrictions. Let us call it the internal type of parameters. More convenient is, probably, the following variant of external real parameters using the angles and side lengths of the trapezoids. The program recognizes that the parameter set is of this second type if there is an angle greater than one degree on a place where the input value of the internal variable must be a ratio less than one. If the input is of the external type, then the data is transformed into the internal parameter type. The optimization is performed, and the final result is transformed back to the external type.

Let us define the external type of parameter input. For a positive $k \leq N$, the parameter d_geom_param[5k-4]=$h_k > 0$ is as before the height of the kth trapezoid measured in μm. The parameter d_geom_param[5k-3]=$b_k - a_k$ is the length of the upper side of the kth trapezoid measured in μm. The real d_geom_param[5k-2] is the interior angle α_k (in degrees) at the right lower corner of the kth trapezoid. The last parameters of the kth trapezoid form the refractive index n_k of the material, i.e. the refractive index is

n_k=d_geom_param[5k-1]+ i*d_geom_param[5k].

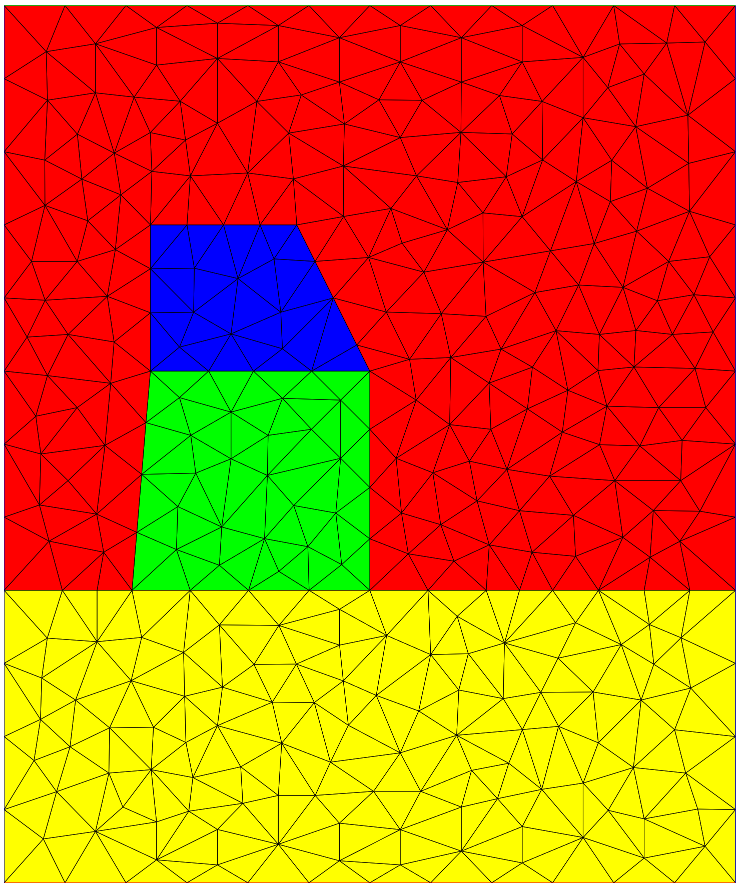
The next parameter d_geom_param[5N+1]=a_0 is the x coordinate (in μm) of the left lower corner of the first trapezoid. The next real parameter d_geom_param[5N+2]=b_0 is the x coordinate (in μm) of the right lower corner of the first trapezoid. Finally, the refractive indices of the cover and substrate material are given as in the internal parameter setting.

In the case of a switch from external to internal parameters, the input of the upper and lower bounds for the angles and upper side lengths of the trapezoids and for the x-coordinates of the lower corner points of the stack are ignored. Instead, the upper and lower bounds for the internal parameters d_geom_param[l] with $l = 5k-3$, $l =$ 5i_geom_param[2]-9 are set to 0.9 and 0.1, respectively. Similarly, the upper and lower bounds for the internal parameters d_geom_param[l] with $l = 5k-2$ and with $l =$ 5i_geom_param[2]-8 are set to 0.9 and 0., respectively.

10.2.6 Optimization of a polygonal interface inside a general grating

The fifth class (i_geom_param[1]=5) is designed to optimize a small detail inside a fixed complex grating geometry. The basis is a general grating defined by an input file "nameG.inp". This is to be extended by introducing a new polygonal interface curve dividing one of the material areas of the given grating into two (cf. Figure 31 where the blue rectangle on the left is split into a blue and yellow part on the right by a polygonal consisting of three segments). The task of optimization is to find the optimal interface among all the polygonal interface curves with fixed end-points dividing the fixed area.

The divided area A is supposed to be convex. Moreover, A must not reach to the upper and lower boundary lines of the grating cross section. The new interface connects two prescribed boundary points P_1 and P_2 of the convex area A which must be in the list of grid points in "nameG.inp". Moreover, the new polygonal interface is sought in form of a graph of a piecewise linear function defined over a uniform partition of the straight-line segment $[P_1, P_2]$. In other words, there is a positive integer N such that the new interface is the polygonal curve connecting the end-points P_1 and P_2 through the corner points Q_1, Q_2, $\dots$,Q_N located in the interior of the convex area A. The orthogonal projections of these



🟥 1.000 +i 0.000
🟩 1.300 +i 0.200
🟦 1.200 +i 0.300
🟨 1.500 +i 0.000

Figure 29: Example grid for stack of trapezoids.

Q_k onto the the straight-line segment $[P_1, P_2]$ are supposed to form a uniform partition of $[P_1, P_2]$. Hence, if ν is the unit normal perpendicular to $[P_2 - P_1]$ and pointing to the left of $[P_1, P_2]$, then the interior corners are given as (cf. Figure 30)

$$Q_k := P_1 + \frac{k}{N+1} [P_2 - P_1] + h_k \nu, \ k = 1, \dots, N, \tag{10.8}$$

where h_k denotes the hight of Q_k over $[P_1, P_2]$.

The refractive indices of the grating materials (except those of the substrate and cover material) are included into the set of optimization operators. A special case of this class (switched on by choosing i_geom_param[3]=0) is the optimization of only the refractive indices without any geometry parameter, namely the optimization of the refractive indices in the fixed grating geometry "nameG.inp".

The number ns_geom_param of name parameters is one and s_geom_param[1] contains the name "nameG" of the file "nameG.inp" without tag ".inp". Recall that this is the

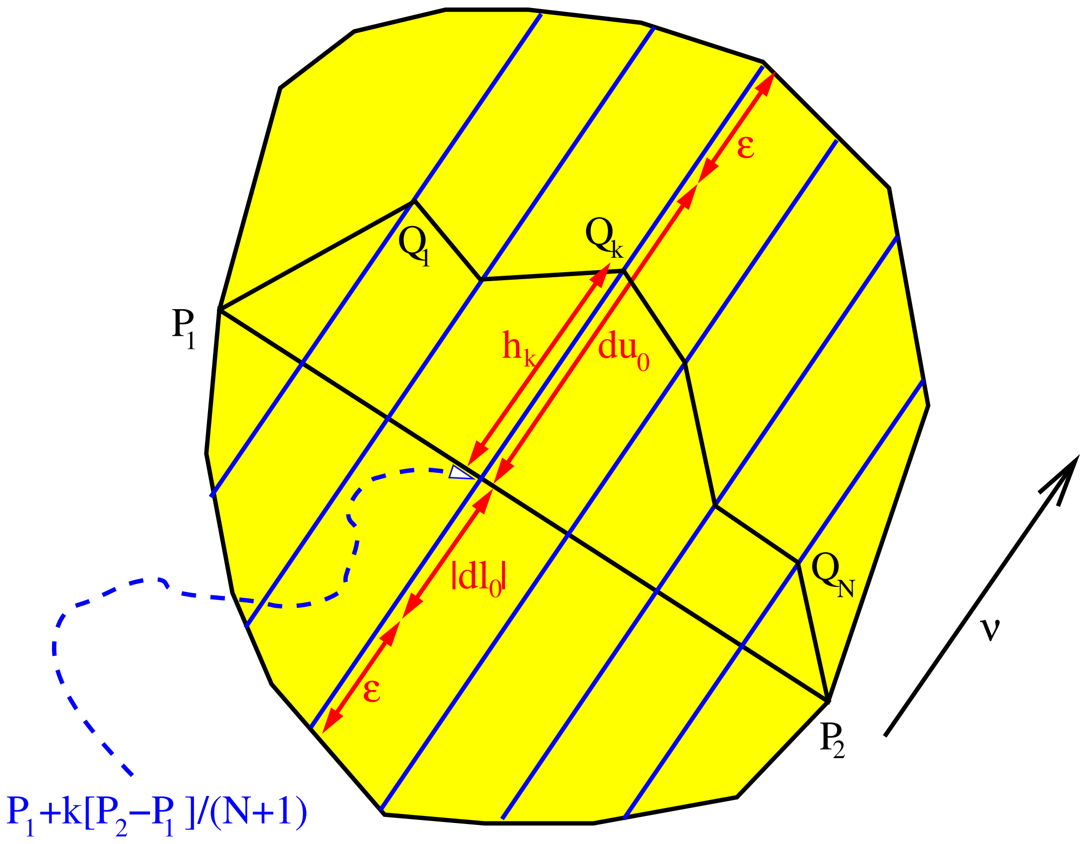


Figure 30: Location of interface in yellow area A.

geometry input file of the grating which is to be extended (cf. Section 3.2 and the example in Section 12.1).

The fifth class requires the following ni_geom_param=6 integer parameters:

i_geom_param[1]=index 5 of the grating class
i_geom_param[2]=number of materials which must be one plus the number of different materials indicated in "nameG.inp" if i_geom_param[3]> 0 and which is exactly the number of different materials indicated in "nameG.inp" if i_geom_param[3]=0
i_geom_param[3]=number N of interior corners in the polygonal interface, non-negative integer
i_geom_param[4]=index of end-point P1 as a grid point in "nameG.inp" resp. dummy if i_geom_param[3]=0
i_geom_param[5]=index of end-point P2 as a grid point in "nameG.inp" resp. dummy if i_geom_param[3]=0
i_geom_param[6]=index of convex domain A which is divided by the new polygonal interface as a subdomain in "nameG.inp" resp. dummy if i_geom_param[3]=0

After dividing the convex domain A by the new interface, the first subdomain on the right of the polygonal interface running from P1 to P2 inherits the material index i_geom_param[6], and the index of the second subdomain is set to i_geom_param[2]-1. Before the subdivision, i_geom_param[2]-1 was the material index of the domain adjacent to the lower boundary line of the grating. Now the index of the domain adjacent to the lower boundary line is changed to i_geom_param[2][24].

The number nd_geom_param of real input parameters is equal to 2i_geom_param[2]+ i_geom_param[3]+1. The first 2i_geom_param[2] of these real valued parameters define the refractive indices n_k of the grating materials occupying the domains with the indices $k = 1, ...,$i_geom_param[2] by

$$n_k\text{=d\_geom\_param[2k-1]+i*d\_geom\_param[2k].}$$

Clearly, the real parts of all refractive indices must be positive and the imaginary parts non-negative. The imaginary part of the cover material must vanish. The parameter d_geom_param[k] with k=2i_geom_param[2]+j, j=1,...,i_geom_param[3] is just the height $h_j > 0$ (cf. Equation (10.8)) in μm of the interior corner point Q_j over the line through P_1 and P_2. Finally, the last of the real input parameters d_geom_param[k] with the index k=2i_geom_param[2]+i_geom_param[3]+1 is the threshold ε for the distance of the interior interface corner points to the boundary of the convex domain A (cf. Figure 30 and the subsequent Equation (10.9)). The distance is measured in μm in the direction of the normal ν to [P1,P2]. Of course, if i_geom_param[3]=0, then the last real parameter is a dummy.

Setting upper bound equal to lower bound, the user must fix the refractive indices of the substrate and cover material and the last threshold parameter, i.e., the equality

$$\begin{gathered}\text{du\_geom\_param}[j]\text{=dl\_geom\_param}[j],\\ j = 1, 2, \text{2i\_geom\_param[2]-1,2i\_geom\_param[2]},\\ \text{i\_geom\_param[2]+i\_geom\_param[3]+1.}\end{gathered}$$

must be satisfied. For j = 2i_geom_param[2]+k with $1 \leq k \leq$i_geom_param[3], the user defined upper and lower bounds du_geom_param[j] and dl_geom_param[j] will be corrected in order to guarantee that the corresponding interior corner points Q_k remain inside the convex domain A. More precisely, if the numbers dl_1=dl_geom_param[j] and du_1=du_geom_param[j] are the old user defined values, then the new internally corrected values dl_2=dl_geom_param[j] and du_2=du_geom_param[j] are given by (cf. Figure 30)

$$\begin{aligned}du_2 &:= \min\{du_1, du_0\},\ du_0 := \sup\left\{h > 0:\ P_1 + \frac{k}{N+1}[P_2 - P_1] + h\nu \in A\right\} - \varepsilon,\\ dl_2 &:= \max\{dl_1, dl_0\},\quad dl_0 := \inf\left\{h < 0:\ P_1 + \frac{k}{N+1}[P_2 - P_1] + h\nu \in A\right\} + \varepsilon,\\ \varepsilon &:= \text{d\_geom\_param}\Big[\text{2i\_geom\_param[2]+i\_geom\_param[3]+1}\Big]. \qquad (10.9)\end{aligned}$$

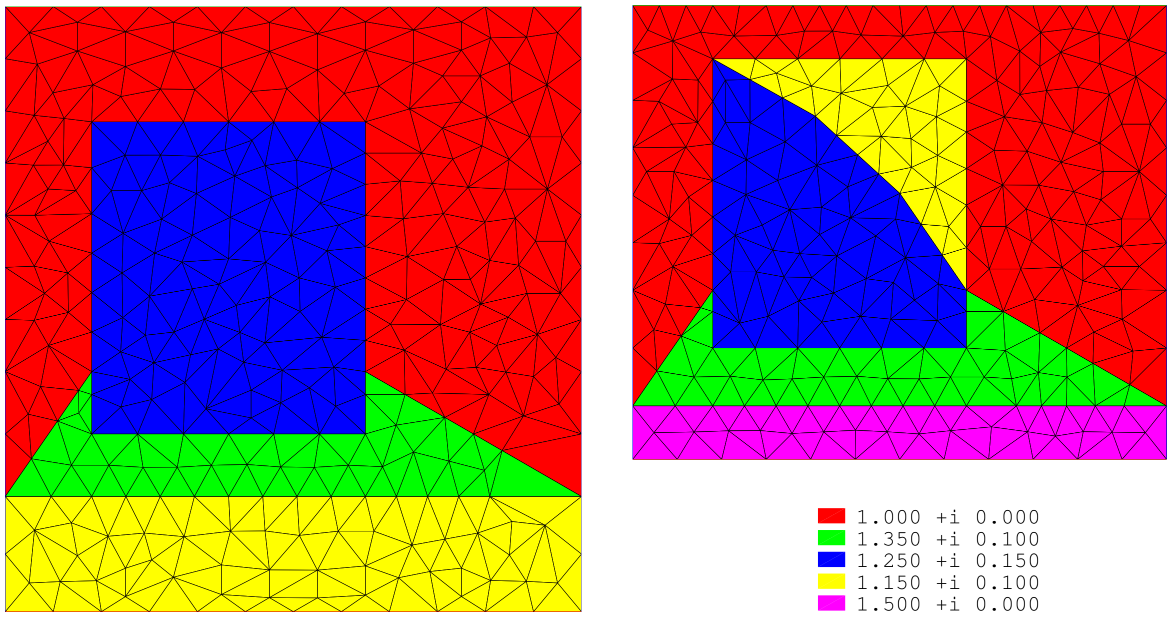


Figure 31: General grating without and with additional interface.

10.2.7 Bridge composed of trapezoids under light in the EUV range

The sixth class (i_geom_param[1]=6) is a flat grating with a bridge in form of a stack of trapezoids put on the flat interface (cf. Figure 32). The trapezoids are located with their parallel sides in the direction of the x-axis, and one trapezoid is placed over the other such that the upper resp. lower sides of the two adjacent trapezoids coincide. All these trapezoids do not exceed the period $\{(x, y) : \ 0 \leq x \leq d\}$. If necessary, additional coating layers beneath the grating structure are allowed. Beside the bridge an extra layer can be added.

The class has no name parameters (ns_geom_param=0) and ni_geom_param=5 integer parameters. The value i_geom_param[2] is the number of different materials which is equal to the number N of trapezoids plus two (two for substrate and cover material) and, eventually, plus one if an extra side layer is added. The number i_geom_param[3] is the number N of trapezoids. The fourth integer parameter i_geom_param[4] is reserved for the number M of lower layers, the refractive indices or widths of which are included into the set of optimization parameters. Finally, i_geom_param[5] is the index of the trapezoid in the bridge intersected by the upper boundary line of the extra layer beside the bridge. If there is no extra layer beside the bridge, then i_geom_param[5]=1 and the height of the layer (cf. the subsequent parameter d_geom_param[nd_geom_param-9]) is zero.

The number of real parameters nd_geom_param is equal to $5N + 3M + 12$. These parameters include five reals for each trapezoid, two reals for the location of the stack of trapezoids, three reals for each layer beneath the bridge, and four reals for the refractive indices of the substrate and cover material. In particular (cf. Figure 32), if k is a positive integer less or equal to N, then the parameter d_geom_param[5k-4]=$h_k > 0$ is the height of

[24] If the indexing should be confusing, then the resulting distribution of the materials can be checked entering "OPTIMIZE -i name.dat". With this result a picture like the right in the Figure 31 appears.

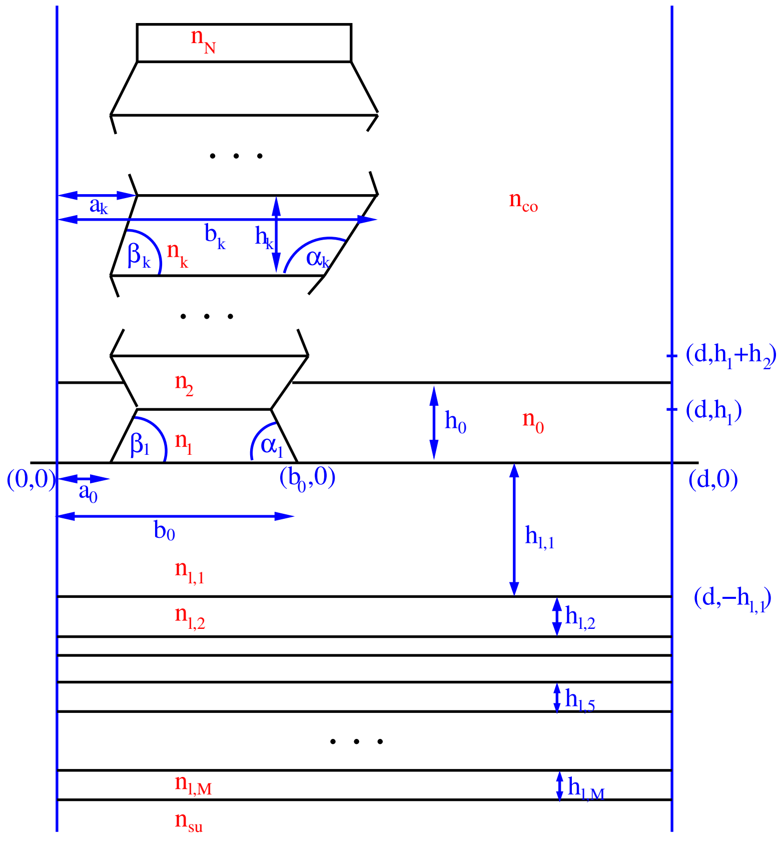


Figure 32: Bridge composed of trapezoids under light in the EUV range .

the kth trapezoid measured in μm. The parameter d_geom_param[5k-3]=$b_k/d \in (0,1]$ is the ratio of the x coordinate of the right upper corner of the kth trapezoid over the period d of the grating. The real d_geom_param[5k-2]= $a_k/b_k \in [0,1)$ is the ratio of the x coordinate of the left upper corner of the kth trapezoid over the x coordinate of the right upper corner. The last parameters of the kth trapezoid form the refractive index n_k of the material, i.e. the refractive index is

n_k=d_geom_param[5k-1]+ i*d_geom_param[5k].

The next parameter d_geom_param[5N+1] is the ratio $b_0/d \in (0,1]$ of the x coordinate of the right lower corner of the first trapezoid over the period d of the grating. The next real parameter d_geom_param[5N+2]=$a_0/b_0 \in [0,1)$ is the ratio of the x coordinate of the left lower corner of the first trapezoid over the x coordinate of the right lower corner. If M¿0, then there follow three reals for each lower layer included into the optimization part. In particular, if k is a positive integer less or equal to M, then the parameter $h_{l,k}$ =d_geom_param[5*N+3+3k] is the height of the kth layer in μm. The refractive index of the corresponding material is

$n_{l,k}$ =d_geom_param[5k-1]+ i*d_geom_param[5k].

The real parameter of index [nd_geom_param-9] is the relative height of the upper boundary line of the extra layer beside the bridge inside the trapezoid of index i_geom_param[5], i.e., it defines the height h_0 of the extra layer beside the bridge by the formula

$$\begin{aligned} h_0 &= \sum_{k=1}^{\text{i\_geom\_param[5]}-1} h_k + h_{\text{rel}} \;, \\ h_{\text{rel}} &= h_{\text{i\_geom\_param[5]}} \cdot \text{d\_geom\_param[nd\_geom\_param} - 9] \;. \end{aligned}$$

The refractive index of the corresponding layer material is

n_0=d_geom_param[nd_geom_param-8]+ i*d_geom_param[nd_geom_param-7].

The refractive indices of the cover and substrate material are given as

n_{co}=d_geom_param[nd_geom_param-6]+i*d_geom_param[nd_geom_param-5],
n_{su}=d_geom_param[nd_geom_param-4]+i*d_geom_param[nd_geom_param-3].

The last three real parameters restrict the sidewall angles α_k and β_k for $k = 1, \dots, N-1$ (cf. Figure 32). For these angles, the real numbers

$$\begin{aligned} \varphi_{min} &:= \text{d\_geom\_param[nd\_geom\_param} - 2], \\ \varphi_{max} &:= \text{d\_geom\_param[nd\_geom\_param} - 1] \end{aligned}$$

are the lower and upper bound, respectively. However, instead of requiring the strict fulfillment of $\varphi_{min} \le \alpha_k \le \varphi_{max}$ and $\varphi_{min} \le \beta_k \le \varphi_{max}$, $k = 1, \dots, N-1$, we shall enforce the bounds only weakly by adding the penalty term

$$\varphi_{fac} \cdot \sum_{k=1}^{N-1} \Big\{ \max\{0, \alpha_k - \varphi_{max}\}^2 + \max\{0, \varphi_{min} - \alpha_k\}^2 + \max\{0, \beta_k - \varphi_{max}\}^2 + \max\{0, \varphi_{min} - \beta_k\}^2 \Big\} \tag{10.10}$$

to the objective functional (10.2). The calibration factor φ_{fac} is the last real parameter d_geom_param[nd_geom_param] and controls the strength of the required boundedness condition.

Setting upper bound equal to lower bound, the user must fix the refractive indices of the substrate and the cover material, i.e., the equality du_geom_param[j]=dl_geom_param[j] must be satisfied for all indices j =nd_geom_param-6, ... ,nd_geom_param-3. Clearly, the real parts of all refractive indices must be positive and the imaginary parts non-negative. The imaginary part of the cover material must vanish. Like the refractive indices of substrate and cover material the last three parameters must be fixed setting upper bound equal to lower bound.

If the user wants to restrict the optimization to gratings with sidewall angles $\alpha_k \le 90°$ and $\beta_k \le 90°$ (cf. Figure 32), then he must set the environment variable `EUV_SWA_90` to "yes". In order to obtain box constraints, this choice requires a change in the meaning of the parameters d_geom_param[i] for $i = 5(j-1)+2$ and $i = 5(j-1)+3$ with $j = 1, 2, \dots, N$. More precisely,

$$\begin{aligned} \text{d\_geom\_param}[5(j-1)+2] &:= \frac{b_j}{b_{j-1}} \\ \text{d\_geom\_param}[5(j-1)+3] &:= \frac{a_j - a_{j-1}}{b_j - a_{j-1}}, \quad j = 1, \dots, N = \text{i\_geom\_param[3]}, \end{aligned}$$

where the numbers a_j and b_j are defined in Figure 32. Clearly, $0 < \alpha_k \leq 90°$ and $0 < \beta_k \leq 90°$ imply the estimate $0 <$d_geom_param[i]≤ 1 for $i = 5(j-1)+2$ and $i = 5(j-1)+3$ with $j = 1, 2, \ldots, N$. Therefore, the corresponding lower and upper bounds must satisfy the relation $0 \leq$ dl_geom_param[i]$\leq$ du_geom_param[i] ≤ 1.

Finally, note that the automatic generation of an additional strip for the FEM domain is switched off (compare Section 3.2) if the classical case of TE polarization is considered (i.e. type of polarization is "TE", angle of incidence ϕ is zero, and output data is presented in the "TE/TM" form of Section 2.3) and if the x-coordinates of the lower corner points of the stack of trapezoids are fixed to period times rational number:

$$\begin{aligned}
\text{du\_geom\_param}[5\cdot N+1] &= \text{dl\_geom\_param}[5\cdot N+1], \\
\text{du\_geom\_param}[5\cdot N+2] &= \text{dl\_geom\_param}[5\cdot N+2], \\
\text{du\_geom\_param}[5\cdot N+1] &= \frac{k_1}{l_1}, \quad k_1, l_1 \in \mathbb{N},\ l_1 < 1\,000 \\
\text{du\_geom\_param}[5\cdot N+2]\cdot\text{du\_geom\_param}[5\cdot N+1] &= \frac{k_2}{l_2}, \\
& \qquad k_2, l_2 \in \mathbb{N},\ l_2 < 1\,000.
\end{aligned}$$

In the case that the generation of an additional strip is switched off, the material of this strip, i.e., the material of the substrate resp. of the adjacent lower layer must not be counted in the number of materials given in parameter i_geom_param[2]. In order to avoid any restriction to rational values, the automatic generation of an additional strip can be enabled by setting the environmental variable `ADD_STRIPS` to yes.

10.3 Numerical methods of optimization

10.3.1 General parameters of optimization algorithm

All the optimization algorithms implemented for `DIPOG`-2.1 are iterative. So the first control parameter is in common. Its the number niter_max>0 of maximal iterations. The level dependent input is described at the end of Section 10.1.3. If the iteration does not stop earlier, then the number of iterations is exactly equal to this number. A reason to stop earlier is, e.g., that an approximative local solution with a sufficient accuracy has been found before. Another reason could be that the gradient computation is inaccurate and the objective functional does not drop in the direction of the negative gradient.

The next parameter ind_opt is the indicator of the numerical method of optimization and must be a number between 1 and 5. In particular, the choice of ind_opt switches to

- ind_opt=1: Conjugate gradient method with projection
- ind_opt=2: Interior point method
- ind_opt=3: Method of augmented Lagrangian
- ind_opt=4: Simulated annealing
- ind_opt=5: Newton type method
- ind_opt=6: Levenberg-Marquardt method

The first three and the last method are gradient based local optimizers. All of these four can deal with the box constraints imposed by the upper and lower bounds of the real parameters.

However, to additional constraints like that for the polygonal profile gratings (cf. Equation (10.7) in Section 10.2.4), only the implementation of the conjugate gradient method has been adapted. Moreover, the method of augmented Lagrangian requires the computation of the objective functional $f(r)$ at parameter sets r outside the class of admissible solutions. Since this is meaningful only for the gratings with profiles defined by the graph of a linear function, the method of augmented Lagrangian applies only to this grating class. The fourth method is a stochastic global optimizer.

Depending on the value of ind_opt, the general optimization procedure requires further ni_opt integer parameters and nd_opt real valued parameters. The input is as indicated in Section 10.1.3. Finally, the performance of the optimization depends on the last input numbers, the scaling factors.

Indeed, without proper scaling the iteration might stop at a parameter set far from a local minimum. For instance, suppose the partial derivative $\partial f/\partial r_1$ is much larger than the other partial derivatives $\partial f/\partial r_j$ with $j = 2, \dots, N$. Then the iterative solution moves in the negative gradient direction, which is more or less the direction of the first component r_1, and the value $\partial f/\partial r_1$ reduces until it is in the order of the discretization error of the gradient computation. This, however, means that the "gradient" at the actual iterative solution has a first component $\partial f/\partial r_1$ dominated by the discretization error. Moreover, this first component dominates the other components $\partial f/\partial r_j$ due to the bad scaling. Moving into such a "gradient" direction will sooner or later end up in a direction of no descent, and the iteration stops. Though the components $\partial f/\partial r_j$ with $j = 2, \dots, N$ are quite accurate, they are not used for a correction of the iterative solution towards the local minimum.

To improve the scaling, the parameters r_j from $[l_j, u_j]$ are internally replaced by the scaled parameters $r'_j := s_j r_j$ from $[s_j l_j, s_j u_j]$. The partial derivative $\partial f/\partial r_j$ turns into $\partial f/\partial r'_j = 1/s_j \, \partial f/\partial r_j$. Choosing the right scaling factors s_j, the partial derivatives $\partial f/\partial r'_j$ can be made to be almost of the same size, and the iteration converges well. The gradients printed after calling `OPTIMIZE` with flag `-f` may be helpful to find the scalar factors s_j. For the notation from Section 10.1.2, recall that the variables r_j are those parameters d_geom_param[j] which are not fixed by setting dl_geom_param[j]=du_geom_param[j]. The corresponding scaling factors s_j are the values d_geom_scal[j]. In the case of fixed real parameters d_geom_param[j] with dl_geom_param[j]=du_geom_param[j], the scaling factor d_geom_scal[j] must be set to one.

10.3.2 Conjugate gradient method with projection

Suppose the optimization problem is to find a local minimum, i.e., to find an admissible vector r_{opt} in $\mathbb{R}^N$ such that $f(r_{opt}) \leq f(r)$ holds at least for any admissible $r \in \mathbb{R}^N$ close to r_{opt}. Here a vector $r \in \mathbb{R}^N$ is called admissible if the coordinates r_i of r satisfy $l_i \leq r_i \leq u_i$ and if the constraint conditions[25] $g_m(r) \leq 0$ are fulfilled for any $m = 1, \dots, M$. The functionals f and g_m are supposed to be continuously differentiable and, possibly, non-linear. The conjugate gradient methods consists of the following Steps (1)-(3):

[25] Obviously, for the classes in the Sections 10.2.2, 10.2.3, 10.2.5, 10.2.6, and 10.2.7, there are no additional constraints, i.e., $M = 0$. In case of the class of Section 10.2.4, the M constraints $g_m(r) \leq 0$ are the inequalities in Equation 10.7.

(1) **Initialization:** Take the initial solution r_0 from the input file provided by the user. Compute $f(r_0)$ and the gradient $\nabla f(r_0)$ of f at r_0. In case that r_0 is an interior point of the set of admissible vectors, then choose the search direction $sd_0 := -\nabla f(r_0)$. If either a component $[r_0]_i$ of r_0 is equal to the bound u_i resp. l_i or if $g_m(r_0) = 0$ holds for some m, then set sd_0 equal to the projection of $-\nabla f(r_0)$ to the tangent cone of directions pointing from r_0 toward admissible points. Note that, if the difference of the component $[r_0]_i$ to the bounds u_i resp. l_i is less than the small prescribed threshold ε_{acc}, then we set $[r_0]_i$ equal to u_i resp. l_i, and the search direction is chosen as for the boundary point[26]. In any case we write $sd_0 := P[-\nabla f(r_0)]$. Set the iteration index j to 0.

(2) **Iteration step:** If the number of iterations j is larger than the prescribed number niter_max, then stop the iteration and go to the next Step (3). Else compute the norm of the reduced gradient $\|P[-\nabla f(r_j)]\|$. If this is less than the prescribed gradient threshold ε_{gra}, then stop the iteration and go to the next Step (3). Also if the reduced gradient is almost unchanged in the last n_{norm} (prescribed integer) iterations, stop the iteration and go to the next Step (3). Almost unchanged means

$$\frac{\|P[-\nabla f(r_j)] - P[-\nabla f(r_{j-l})]\|}{\|P[-\nabla f(r_j)]\|} \leq \varepsilon_{norm}, \quad l = 1, \dots, n_{norm}$$

with a prescribed small number ε_{norm}. If the stopping conditions fail, then perform a line search (2.1)-(2.2):

(2.1) Initialize $\alpha = \alpha_{max}$.

(2.2) Set $r_{j+1} = r_j + \alpha\, sd_j$. If the difference between r_{j+1} and r_j is less than the machine accuracy, stop and jump to Step (3). Else compute $f(r_{j+1})$ and $\nabla f(r_{j+1})$. If r_{j+1} is admissible and if the Armijo criterion

$$f(r_{j+1}) - f(r_j) \leq c_1\, \alpha\, \langle \nabla f(r_j), sd_j \rangle \tag{10.11}$$

is fulfilled, then stop and leave the line search. Else compute a smaller α by halving the actual value or by a clever approximate minimization of the univariate function $\tilde{\alpha} \mapsto f(r_j + \tilde{\alpha}\, sd_j)$. Repeat Step (2.2).

Determine a new search direction by the following non-linear conjugate gradient formula

$$\begin{aligned} sd_{j+1} &:= -P\left[\nabla f(r_{j+1}) + \beta\, sd_j\right], \\ \beta &:= \max\left\{0, \frac{\langle \nabla f(r_{j+1}), \nabla f(r_{j+1}) - \nabla f(r_j)\rangle}{\langle \nabla f(r_j), \nabla f(r_j)\rangle}\right\}. \end{aligned} \tag{10.12}$$

Set $j = j + 1$ and repeat Step (2).

[26] This change of r_0 and the corresponding change of the subsequent iterates r_j before the application of the projection of $P[-\nabla f(r_0)]$ avoids a lot of unnecessary time consuming tiny iteration steps toward the boundary.

(3) **Final output:** Accept and print r_j as the local solution. Print the corresponding values of $f(r_j)$ and $\|P[\nabla f(r_j)]\|$.

The conjugate gradient method (ind_opt=1) requires one integer input value (ni_opt=1). This i_opt[1] is the threshold integer $n_{norm} > 1$ (choose, e.g., i_opt[1]=3). Whenever the gradient of the iterative solution remains unchanged over n_{norm} iterations, then the iteration is stopped (cf. Step (2)). The number of real parameters is nd_opt=5. These parameters are:

- d_opt[1]: Maximal stepsize factor $\alpha_{max} > 0$ in line search (cf. Step (2.1) and choose, e.g., $\alpha_{max} = 1$)
- d_opt[2]: Constant c_1, $0 < c_1 < 1$ in Armijo stopping criterion (10.11) for line search (e.g. $c_1 = 0.001$)
- d_opt[3]: Threshold $\varepsilon_{acc} > 0$, iterate is shifted to the boundary if the distance to the boundary is less than ε_{acc} (cf. the projection P in Step (1) and choose ε_{acc} about the expected accuracy of the final solution)
- d_opt[4]: Threshold $\varepsilon_{gra} > 0$, stop if gradient is less than ε_{gra} (should be about approximation error of gradient or less)
- d_opt[5]: Threshold $\varepsilon_{norm} > 0$, stop if relative change of the norm of the gradient is less than ε_{norm} for the last n_{norm} steps (cf. Step (2) and choose, e.g., $\varepsilon_{norm} = 0.01$)

The parameters i_opt[1] and d_opt[i], $i = 1, 2, 3, 5$ should be chosen as recommended. For the error d_opt[4]$\sim \varepsilon_{gra}$ of the gradient computation, a first estimate can be obtained using the command `OPTIMIZE -g name.dat`. On the other hand, the user can choose niter_max larger than necessary and the positive parameter d_opt[4] smaller than recommended. In the worst case, a large number of unnecessary iteration steps with very small changes in the iterative solutions are performed at the end of the optimization procedure. Even these redundant iteration steps can be interrupted. Indeed, the user can invoke the optimization by the command `OPTIMIZE -g name.dat`. In this case the actual iterative solutions are printed on the screen, and, pushing the `Ctrl C` keys, the user can stop the iteration whenever he observes that the iterative solutions do not improve.

10.3.3 Interior point method

Suppose the optimization problem is to find a local minimum, i.e., to find an admissible vector r_{opt} in $\mathbb{R}^N$ such that $f(r_{opt}) \leq f(r)$ holds at least for any admissible $r \in \mathbb{R}^N$ close to r_{opt}. Here a vector $r \in \mathbb{R}^N$ is called admissible if the coordinates r_i of r satisfy $l_i \leq r_i \leq u_i$. The functional f is continuously differentiable and, possibly, non-linear.

To prepare the interior point method, some definitions are needed. The slack variables sl and su as well the dual slack variables dl and du are given by

$$sl := (sl_i)_{i=1}^N, \; sl_i := r_i - l_i \geq 0, \; dl := (dl_i)_{i=1}^N, \; dl_i \geq 0, \tag{10.13}$$

$$su := (su_i)_{i=1}^N, \; su_i := u_i - r_i \geq 0, \; du := (du_i)_{i=1}^N, \; du_i \geq 0. \tag{10.14}$$

Introducing the operator $F_\varrho : \ (r, sl, su, dl, du) \mapsto F_\varrho(r, sl, su, dl, du) \in \mathbb{R}^{5N}$ by

$$F_\varrho(r, sl, su, dl, du) \ := \begin{pmatrix} du - dl + \nabla f(r) \\ r - l - sl \\ u - r - su \\ (sl_i\, dl_i - \varrho 1)_{i=1}^N \\ (su_i\, du_i - \varrho 1)_{i=1}^N \end{pmatrix},$$

the necessary Karush-Kuhn-Tucker condition of a locally optimal solution imply that there exist dual slack variables dl and du such that

$$F_0(r, sl, su, dl, du) = 0,\ sl \geq 0,\ su \geq 0,\ dl \geq 0,\ du \geq 0.$$

Finally, the interior point method defines iterative solutions[27] r_j as the approximate solution of the equations $F_{\varrho_j}(r_j, sl_j, su_j, dl_j, du_j) = 0$, where $\varrho_j = \varrho_0\, q^j$ with $0 < q < 1$ tends exponentially to zero and where the approximate solution $(r_j, sl_j, su_j, dl_j, du_j)$ is obtained by one step of Newton's method choosing $(r_{j-1}, sl_{j-1}, su_{j-1}, dl_{j-1}, du_{j-1})$ as initial solution.

More precisely, the interior point method consists of the following Steps (1)-(3):

(1) **Initialization:** Set the index j of the iteration to zero, choose r_0 from the initial values given by the user in the input file, and suppose the strict fulfillment of the restrictions $l_i < [r_0]_i < u_i,\ i = 1, \ldots, N$. Define sl_0 and su_0 using the Equations (10.13) and (10.14). Read the initial value ϱ_0 from the input of the user and introduce the dual slack variables dl_0 and du_0 by $[dl_0]_i := \varrho_0/[sl_0]_i$ and $[du_0]_i := \varrho_0/[su_0]_i$. Increase j by setting $j = 1$.

(2) **Iteration step:** If the number of iterations j is larger than the prescribed number niter_max, then stop the iteration and go to the next Step (3). Else compute $f(r_j)$ and the gradient $\nabla f(r_j)$. Determine the reduced gradient $\mathcal{P}(\nabla f(r_j))$ by setting to zero all those components of $\nabla f(r_j)$ for which $[r_j]_i < l_i + \varepsilon_{acc}$ and $[\nabla f(r_j)]_i > 0$ and for which $[r_j]_i > u_i - \varepsilon_{acc}$ and $[\nabla f(r_j)]_i < 0$. Here ε_{acc} is a small prescribed threshold. If $\|\mathcal{P}(\nabla f(r_j))\|$ is less than the prescribed small constant ε_{gra}, then stop the iteration and go to the next Step (3). Also if the reduced gradient is almost unchanged in the last n_{norm} (prescribed integer) iterations, then stop the iteration and go to the next Step (3). Almost unchanged means

$$\frac{\|\mathcal{P}(\nabla f(r_j)) - \mathcal{P}(\nabla f(r_{j-l}))\|}{\|\mathcal{P}(\nabla f(r_j))\|} \leq \varepsilon_{norm}, \quad l = 1, \ldots, n_{norm} \tag{10.15}$$

with a prescribed small number ε_{norm}. Now suppose none of the stopping conditions is satisfied. For prescribed $\varrho_0 > 0$ and q with $0 < q < 1$, introduce the parameter $\varrho_j = \varrho_0\, q^j$. Compute the search direction of the step $(sdr_j, sdsl_j, sdsu_j, sddl_j, sddu_j)$ by solving $F_{\varrho_j}(sdr_j, sdsl_j, sdsu_j, sddl_j, sddu_j) = 0$ approximately by the Newton step

$$(sdr_j, sdsl_j, sdsu_j, sddl_j, sddu_j) =$$
$$- \left[\nabla F_{\varrho_j}\left(r_j, sl_j, su_j, dl_j, du_j\right)\right]^{-1} F_{\varrho_j}\left(r_j, sl_j, su_j, dl_j, du_j\right).$$

[27]Take heed that a symbol with lower index i denotes the ith component of a vector whereas the same symbol with lower index j denotes the jth iterate of the vector in an iterative process.

Note that the computation of ∇F_{ϱ_j} requires the computation of the Hessian $\nabla^2 f(r_j)$. This is approximated using the Broyden-Fletcher-Goldfarb-Shanno update which is based on first order derivatives, only. Now, knowing the search direction, perform the line search (2.1)-(2.2):

(2.1) Initialize $\alpha = \alpha_{max}$. Eventually, reduce α such that $sl_j + \alpha\, sdsl_j > 0$, $su_j + \alpha\, sdsu_j > 0$, $dl_j + \alpha\, sddl_j > 0$, and $du_j + \alpha\, sddu_j > 0$ hold.

(2.2) Set $r_{j+1} = r_j + \alpha\, sdr_j$, $sl_{j+1} = sl_j + \alpha\, sdsl_j$, $su_{j+1} = su_j + \alpha\, sdsu_j$, $dl_{j+1} = dl_j + \alpha\, sddl_j$, and $du_{j+1} = du_j + \alpha\, sddu_j$. If the difference $\|r_{j+1} - r_j\|$ is less than the prescribed small positive ε_{ste}, stop and jump to Step (3). Else compute $f(r_{j+1})$, $F_{\varrho j+1}(r_{j+1}, sl_{j+1}, su_{j+1}, dl_{j+1}, du_{j+1})$ and $\nabla f(r_{j+1})$. If the Armijo criterion

$$f(r_{j+1}) - f(r_j) \le c_1\, \alpha\, \langle \nabla f(r_j), sdr_j \rangle \qquad (10.16)$$

is fulfilled, then stop and leave the line search. Else compute a smaller α by halving its value or by a clever approximate minimization of the univariate function $\tilde{\alpha} \mapsto f(r_j + \tilde{\alpha}\, sdr_j)$. Repeat Step (2.2).

Finally set $j = j + 1$ and repeat the Step (2).

(3) **Final output:** Accept and print r_j as the local solution. Print the corresponding values of $f(r_j)$ and $\|\mathcal{P}[\nabla f(r_j)]\|$.

The interior point method (ind_opt=2) requires one integer input value (ni_opt=1). This i_opt[1] is the threshold integer $n_{norm} > 1$ (choose, e.g., i_opt[1]=3). Whenever the gradient of the iterative solution remains unchanged over n_{norm} iterations, then the iteration is stopped (cf. Equation (10.15)). The number of real parameters is nd_opt=8. These parameters are:

- d_opt[1]: Initial value $\varrho_0 > 0$ for parameter of operator F_ϱ (cf. Step (2) and choose, e.g., $\varrho_0 = 0.1$)
- d_opt[2]: Reduction factor $q,\ 0 < q < 1$ to reduce parameter ϱ_j of operator F_ϱ (cf. Step (2) and choose, e.g., $q = 0.5$)
- d_opt[3]: Constant $c_1,\ 0 < c_1 < 1$ in Armijo stopping criterion (10.16) for line search (choose, e.g., $c_1 = 0.001$)
- d_opt[4]: Maximal stepsize factor α_{max} ($0 < \alpha_{max} < 1$) in line search (cf. Step (2.1) and choose, e.g., $\alpha_{max} = 0.9$)
- d_opt[5]: Threshold $\varepsilon_{acc} > 0$, iterate is treated as a boundary point if its distance to the boundary is less than ε_{acc} (cf. Step (2) and choose its value about the expected accuracy of the final solution)
- d_opt[6]: Threshold $\varepsilon_{ste} \ge 0$, stop if the change in the iterative solution is less than ε_{ste} (cf. Step (2.2) and choose, e.g., $\varepsilon_{ste} = 0.1\,\varepsilon_{acc}$)
- d_opt[7]: Threshold $\varepsilon_{gra} \ge 0$, stop if gradient is less than ε_{gra} (cf. Step (2) and choose its value about the discretization error of gradient calculation or less)
- d_opt[8]: Threshold $\varepsilon_{norm} > 0$, stop if relative change of the norm of the gradient is less than ε_{norm} for the last n_{norm} steps (cf. Equation (10.15) and choose, e.g., $\varepsilon_{norm} = 0.01$)

The parameters i_opt[1] and d_opt[i], $i = 1, 2, 3, 4, 5, 8$ should be chosen as recommended. For the error d_opt[7]$\sim \varepsilon_{gra}$ of the gradient computation, a first estimate can be obtained using the command `OPTIMIZE -g name.dat`. On the other hand, the user can choose niter_max larger than necessary and the positive parameters d_opt[i], $i = 6, 7$ smaller than recommended. In the worst case, a large number of unnecessary iteration steps with very small changes in the iterative solutions are performed at the end of the optimization procedure. Even these redundant iteration steps can be interrupted. Indeed, the user can invoke the optimization by the command `OPTIMIZE -g name.dat`. In this case the actual iterative solutions are printed on the screen, and, pushing the `Ctrl C` keys, the user can stop the iteration whenever he observes that the iterative solutions do not improve.

10.3.4 Method of augmented Lagrangian

Suppose the optimization problem is to find a local minimum, i.e., to find an admissible vector r_{opt} in $\mathbb{R}^N$ such that $f(r_{opt}) \leq f(r)$ holds at least for any admissible $r \in \mathbb{R}^N$ close to r_{opt}. Here a vector $r \in \mathbb{R}^N$ is called admissible if the coordinates r_i of r satisfy $l_i \leq r_i \leq u_i$. The functional f is continuously differentiable and, possibly, non-linear. Moreover, $f(r)$ is supposed to be defined for all $r \in \mathbb{R}^N$.

To prepare the method of augmented Lagrangian, some definitions are needed. The Lagrangian multipliers are denoted by $ml \in \mathbb{R}^N$ and $mu \in \mathbb{R}^N$, and the augmented Lagrangian is defined by

$$\begin{aligned}\mathcal{L}_\varrho(r, ml, mu) \;:=\; & c_{cal}\, f(r) + \frac{1}{2\varrho} \sum_{i=1}^{N} \left[\max\left\{0, mu_i + \varrho(r_i - u_i)\right\}^2 - mu_i^2\right] \\ & + \frac{1}{2\varrho} \sum_{i=1}^{N} \left[\max\left\{0, ml_i + \varrho(l_i - r_i)\right\}^2 - ml_i^2\right]. \end{aligned} \qquad (10.17)$$

Here ϱ stands for a prescribed positive real parameter and c_{cal} is a fixed calibration factor. Using this notation, the method of augmented Lagrangian consists of the following Steps (1)-(3):

(1) **Initialization:** Set the index j of the iteration to zero and choose r_0 from the initial values given by the user in the input file. Choose the initial multipliers $mu_0 = 0$ and $ml_0 = 0$. Increase j by one.

(2) **Step of iteration:** If the number of iterations j is larger than the prescribed number niter_max, then stop the iteration and go to the next Step (3). Else determine the next iterate r_{j+1} as the optimal vector in $\mathbb{R}^N$ for which the Lagrangian $r \mapsto \mathcal{L}(r, ml_j, mu_j)$ attains its minimum:

> This optimization problem without any restriction is solved by the non-linear conjugate gradient method (inner iteration) and by choosing r_j as the initial solution. In particular, in each inner iteration a new search direction is sought (cf. (10.12) and replace the projection P by the identity), and a line search is performed in this direction (cf. Steps (2.1)-(2.2) in Section 10.3.2). For the line search, a fixed parameter α_{max} for the maximal stepsize factor and a constant

c_1 for the Armijo criterion are needed. Moreover, the maximal number of inner conjugate gradient iteration is bounded by the prescribed number liter_max. The inner iteration stops even earlier if the norm of the gradient of the Lagrangian is less than the prescribed threshold ε_{gra} times c_{cal} or if the stepsize in the line search is less than ε_{acc}.

After the conjugate gradient iteration is finished, define new multipliers $ml_{j+1} = ([ml_{j+1}]_i)_{i=1}^N$ and $mu_{j+1} = ([mu_{j+1}]_i)_{i=1}^N$ by

$$\begin{aligned} [ml_{j+1}]_i &:= \max\Big\{0, [ml_j]_i + \varrho(l_i - [r_{j+1}]_i)\Big\}, \\ [mu_{j+1}]_i &:= \max\Big\{0, [mu_j]_i + \varrho([r_{j+1}]_i - u_i)\Big\}. \end{aligned}$$

If the norm difference $\|ml_{j+1} - ml_j\| + \|mu_{j+1} - mu_j\|$ of old and new multipliers is less than the prescribed positive threshold ε_{mul} and if $\|\nabla_r \mathcal{L}(r_{j+1}, ml_{j+1}, mu_{j+1})\|$ is less than the prescribed gradient threshold ε_{gra}, then stop and go to Step (3). Also if the gradient is almost unchanged in the last n_{norm} (prescribed integer) iterations, then stop the iteration and go to the next Step (3). Almost unchanged means that

$$\frac{\|\nabla_r \mathcal{L}(r_{j+1}, ml_{j+1}, mu_{j+1}) - \nabla_r \mathcal{L}(r_{j+1-l}, ml_{j+1-l}, mu_{j+1-l})\|}{\|\nabla_r \mathcal{L}(r_{j+1}, ml_{j+1}, mu_{j+1})\|} \leq \varepsilon_{norm} \qquad (10.18)$$

holds for $l = 1, \ldots, n_{norm}$ with a prescribed small number ε_{norm}. If all the stopping criteria fail, then increase j by one and repeat Step (2).

(3) **Final output:** Accept and print r_j as the local solution. Determine the reduced gradient $\mathcal{P}(\nabla f(r_j))$ by setting to zero all those components of $\nabla f(r_j)$ for which $[r_j]_i < l_i + \varepsilon_{acc}$ and $[\nabla f(r_j)]_i > 0$ and for which $[r_j]_i > u_i - \varepsilon_{acc}$ and $[\nabla f(r_j)]_i < 0$. Print the corresponding values of $f(r_j)$ and $\|\mathcal{P}[\nabla f(r_j)]\|$.

The method of augmented Lagrangian (ind_opt=3) requires ni_opt=2 integer input values. The first integer parameter i_opt[1] is the maximal number liter_max>0 of conjugate gradient steps in the inner iteration of Step (2). The second i_opt[2] is the threshold integer $n_{norm} > 0$ (choose, e.g., i_opt[1]=3). Whenever the gradient of the iterative solution remains unchanged over n_{norm} iterations, then the iteration is stopped (cf. Step (2)). The number of real parameters is nd_opt=8. These parameters are:

- d_opt[1]: Parameter value $\varrho > 0$ in augmented Lagrangian (cf. (10.17) and choose, e.g., $\varrho = 0.5$)
- d_opt[2]: Calibration factor $c_{cal} > 0$ of objective functional in modified Lagrangian (cf. (10.17) and choose c_{cal} such that c_{cal} times the objective functional is less than one)
- d_opt[3]: Threshold $\varepsilon_{mul} \geq 10^{-13}$, iteration stops if deviation of iterative multipliers is less than ε_{mul} (cf. Step (2) and choose its value about the maximum of i) the desired accuracy of the constraint conditions and ii) the desired accuracy of the minimum value

of the objective functional multiplied by c_{cal})

- d_opt[4]: Threshold $\varepsilon_{gra} \geq 10^{-10}$, inner iteration stops if norm of gradient of Lagrangian is less than $\varepsilon_{gra} \times c_{cal}$ (should be about discretization error of gradient or less)
- d_opt[5]: Threshold $\varepsilon_{acc} \geq 10^{-13}$, inner iteration stops if stepsize in line search is less than ε_{acc}, iterative solution is considered to be at the boundary if its distance to the boundary is less than ε_{acc} (cf. Step (3) and choose, e.g., $\varepsilon_{acc} = 10^{-14}$)
- d_opt[6]: Constant $c_1,\ 0 < c_1 < 1$ in Armijo stopping criterion for line search in inner conjugate gradient iteration (cf. Step (2) and choose, e.g., $c_1 = 0.001$)
- d_opt[7]: Maximal stepsize factor $\alpha_{max} > 0$ in line search (cf. Step (2) and choose, e.g., $\alpha_{max} = 1$)
- d_opt[8]: Threshold $\varepsilon_{norm},\ 0 < \varepsilon_{norm} \leq 1$, stop if relative change in norm of gradient is less than ε_{norm} for the last n_{norm} steps (cf. Equation (10.18) and choose, e.g., $\varepsilon_{norm} = 0.01$)

The parameters i_opt[2] and d_opt[i], $i = 1, 2, 6, 7, 8$ should be chosen as recommended. For the error d_opt[4]$\sim \varepsilon_{gra}$ of the gradient computation, a first estimate can be obtained using the command `OPTIMIZE -g name.dat`. On the other hand, the user can choose niter_max and i_opt[1] larger than necessary and the positive parameters d_opt[i], $i = 3, 4, 5$ smaller than recommended. In the worst case, a large number of unnecessary iteration steps with very small changes in the iterative solutions are performed. The redundant iteration steps of the outer iteration, however, can be interrupted. Indeed, the user can invoke the optimization by the command `OPTIMIZE -g name.dat`. In this case the actual iterative solutions are printed on the screen, and, pushing the `Ctrl C` keys, the user can stop the iteration whenever he observes that the iterative solutions do not improve.

10.3.5 Simulated annealing

Suppose the optimization problem is to find a minimum, i.e., to find an admissible vector r_{opt} in $\mathbb{R}^N$ such that $f(r_{opt}) \leq f(r)$ holds for any admissible $r \in \mathbb{R}^N$. Here a vector $r \in \mathbb{R}^N$ is called admissible if the coordinates r_i of r satisfy $l_i \leq r_i \leq u_i$ and if the constraint conditions $g_m(r) \leq 0$ are fulfilled for any $m = 1, \ldots, M$. The functionals f and g_m are continuously differentiable and, possibly, non-linear. Simulated annealing consists of the following Steps (1)-(5):

(1) **Initialization of restarts:** Set the actual number J of restarts to zero. Read the user defined value of the initial temperature t_{ini}, that of the neighbourhood radius ϱ_{ini}, and the user supplied initial solution $r_{ini} \in \mathbb{R}^N$ from the data file. The temperature must be positive. If $t_{ini} = 0$, then an automatic choice of temperature is provided. Set the first iterate r_0 to r_{ini} and set the first values of the optimal solution r_{opt} to r_{ini}.

(2) **Initialization of iteration:** Set the index j of the iteration step to zero. Set the temperature t to t_{ini} and the value of the neighbourhood radius ϱ to ϱ_{ini}.

(3) **Steps of iteration:** If the index j of the iteration is larger than the prescribed maximal number niter_max, then stop the iteration and go to Step (4). Also if $0 < \|r_j - r_{j-1}\| \leq \varepsilon_{stop}$ with a prescribed threshold ε_{stop}, stop the iteration and go to Step (4). Else get a new admissible iterate r_{j+1} by a random search in the neighbourhood of r_j of radius ϱ (interpreted as a transition of the state in a cooling step). If $f(r_{j+1}) \leq f(r_j)$, then accept r_{j+1} and if $f(r_{j+1}) > f(r_j)$ holds, then accept r_{j+1} with a probability of $\exp(-[f(r_{j+1}) - f(r_j)]/t)$. In the case that r_{j+1} is not accepted, set $r_{j+1} = r_j$. If $f(r_{j+1}) < f(r_{opt})$, then mark the solution setting $r_{opt} = r_{j+1}$. Cool the temperature by multiplying t with the prescribed cooling factor c_{fact}. Decrease the radius of the neighbourhood ϱ by multiplying ϱ with the prescribed neighbourhood reduction factor ϱ_{fact}. Increase index j by one and repeat Step (3).

(4) **Restarts:** If J is greater than the prescribed integer n_{rest}, then generate at random a new starting solution r_0 and go back to Step (2).

(5) **Final output:** Accept and print r_{opt} as the final solution. Print the corresponding value $f(r_{opt})$. For comparison, print the last iterative solution r_j and the corresponding value $f(r_j)$.

The method of simulated annealing (ind_opt=4) requires one integer input value (ni_opt =1). This i_opt[1] is the number of restarts $n_{rest} \geq 0$. The number of real parameters is nd_opt=5. These parameters are:

- d_opt[1]: Initial temperature $t_{ini} > 0$, (chose t_{ini} equal to the variation of the objective functional or set $t_{ini} = 0$ to invoke an automatic choice of t_{ini})
- d_opt[2]: Cooling factor c_{fact}, $0.5 \leq c_{fact} < 1$, in each iteration step the temperature is multiplied by c_{fact}, a slower logarithmic cooling scheme is applied if $c_{fact} = -1$ (choose, e.g., $c_{fact} = 0.95$)
- d_opt[3]: Stopping threshold ε_{stop}, $0 \leq \varepsilon_{stop} < 1$, algorithm stops if the difference of the values of the objective function at actual and previous step differ by a value greater than zero but less than ε_{stop} (choose, e.g., $\varepsilon_{stop} = 0$)
- d_opt[4]: Initial value $\varrho_{ini} > 0$ of radius $\varrho = \varrho_{ini} \min_i[u_i - l_i]$ of neighbourhood where the random search for a new iterate is performed (choose $\varrho = 1$)
- d_opt[5]: Reduction factor ϱ_{fact}, $0 < \varrho_{fact} \leq 1$ to reduce the radius of neighbourhood in each step multiplying ϱ by ϱ_{fact} (choose, e.g., ϱ_{fact} to be the square root of the cooling factor c_{fact})

The maximal number of iterations niter_max should be chosen as large as possible. In other words, the computing time the user is willing to spent determines the choice of niter_max. The parameters d_opt[i], $i = 1, 3, 4, 5$ should be chosen as recommended. The cooling factor d_opt[2] should be chosen sufficiently large such that a lot of random choices (transitions) are accepted at the beginning of the iterative process. However, d_opt[2] should not be too large such that, at the end of the iterative process, the random choices for the new iterative solutions are almost always rejected. Of course, this so called "freezing of the state" should happen at the end, only. If the iterative solution remains unchanged for the last half of the cooling steps, then the cooling factor should be increased. Finally, the number of restarts i_opt[1] should be as large as possible. Indeed, normally, the computation of the objective function is time consuming. Consequently, the values of niter_max and d_opt[2] are, usually, smaller than what the theoretical analysis requires. Using these fast cooling schemes, simulated annealing computes rather a locally optimal solution than a global. Therefore, the restarts will increase the probability that the final solution is the global optimum. Again, i_opt[1] is restricted by the admissible computing time.

The program of simulated annealing can be used for a pure random search. In this case, the parameters should be, e.g.:

i_opt[1] =0
d_opt[1]=10^{20}
d_opt[2]=0.9999999999999
d_opt[3]=0
d_opt[4]=2
d_opt[5]=1

With these parameters, the objective functional is computed at niter_max randomly chosen different parameter sets, and the minimum of these values is determined. Normally, such a random search yields better results than a deterministic search at the points of a regular mesh of the parameter domain.

10.3.6 Newton type method with projection

Suppose the optimization problem is to find a local minimum, i.e., to find an admissible vector r_{opt} in $\mathbb{R}^N$ such that $f(r_{opt}) \leq f(r)$ holds at least for any admissible $r \in \mathbb{R}^N$ close to r_{opt}. Here a vector $r \in \mathbb{R}^N$ is called admissible if the coordinates r_i of r satisfy $l_i \leq r_i \leq u_i$. Moreover suppose the functional f is supposed to be composed of quadratic terms, only (cf. (10.2)). In this case f takes the form

$$f(r) = \| c - \Phi(r) \|^2$$

where Φ maps the admissible parameter sets r into the space $\mathbb{R}^M$ with $M \geq N$, where $c \in \mathbb{R}^M$, and where $\| \cdot \|$ is the Euclidean norm. Clearly, the components of $\Phi(r)$ are just the scaled efficiencies, energies or phase shift values for the grating determined by the parameter set r, and the c is the vector of the scaled prescribed values in (10.2). The scaling factors are the square roots of the positive weights in (10.2).

Denoting the Fréchet derivative of Φ at r by $\nabla\Phi(r)$, we observe

$$\begin{aligned} c \;=\; \Phi\left(r_{opt}\right) \;=\; \Phi\Big(r + [r_{opt} - r]\Big) \;&\sim\; \Phi\left(r\right) + \nabla\Phi\left(r\right) \cdot [r_{opt} - r] \;, \\ \nabla\Phi\left(r\right) \cdot [r_{opt} - r] \;&\sim\; c - \Phi\left(r\right) \;, \\ [r_{opt} - r] \;&\sim\; \Big[\nabla\Phi\left(r\right)^* \nabla\Phi\left(r\right)\Big]^{-1} \nabla\Phi\left(r\right)^* \left[c - \Phi\left(r\right)\right] \;. \end{aligned}$$

This suggests to determine approximate solutions $r_j,\ j = 1, 2, 3, \ldots$ by the following iteration scheme of the Gauß-Newton method. Choose an initial solution r_0 and, for any given

r_j, define r_{j+1} by

$$r_{j+1} := r_j + \Delta r_j, \quad \Delta r_j \;:=\; \Big[\nabla\Phi\,(r_j)^*\,\nabla\Phi\,(r_j)\Big]^{-1}\nabla\Phi\,(r_j)^*\,[c - \Phi\,(r_j)]\;.$$

In the Newton type algorithm of `DIPOG-2.1` the correction term Δr_j is slightly modified in order to guarantee that the new iterate satisfies the constraints $l_i \le r_i \le u_i$. More precisely, if the components of the new iterate r_{j+1} do not fall into the the interval $[l_i, u_i],\ i = 1, \dots, N$, the r_{j+1} is determined as the optimal solution of the following quadratic problem with box constraints:

$$\begin{array}{l} \Big\|\nabla\Phi\,(r_j)\cdot[r_{j+1} - r_j] \;-\; [c - \Phi\,(r_j)]\Big\|^2 \;\longrightarrow\; \min \\ r_{j+1}:\ l_i \le [r_{j+1}]_i \le u_i,\ i = 1, \dots, N \end{array}$$

Obviously, this is a modification of the Gauß-Newton method in the spirit of the Sqp methods.

The Newton type method (ind_opt=5) requires one integer input value (ni_opt=2). The i_opt[1] is the threshold integer $n_{norm} > 1$ (choose, e.g., i_opt[1]=3). Whenever the gradient of the iterative solution remains unchanged over n_{norm} iterations, then the iteration is stopped. The number i_opt[2] is maximal number of iteration for which an increase of the objective functional is accepted (choose, e.g., i_opt[2]=5). The number of real parameters is nd_opt=3. These parameters are:

- d_opt[1]: Threshold $\varepsilon_{acc} > 0$, iterate is shifted to the boundary if the distance to the boundary is less than ε_{acc} (choose ε_{acc} about the expected accuracy of the final solution)
- d_opt[2]: Threshold $\varepsilon_{gra} > 0$, stop if gradient is less than ε_{gra} (should be about approximation error of gradient or less)
- d_opt[3]: Threshold $\varepsilon_{norm} > 0$, stop if relative change of the norm of the gradient is less than ε_{norm} for the last n_{norm} steps (choose, e.g., $\varepsilon_{norm} = 0.01$)

The parameters i_opt[1] and d_opt[i], $i = 1, 3$ should be chosen as recommended. For the error d_opt[2]$\sim \varepsilon_{gra}$ of the gradient computation, a first estimate can be obtained using the command `OPTIMIZE -g name.dat`. On the other hand, the user can choose niter_max larger than necessary and the positive parameter d_opt[2] smaller than recommended. In the worst case, a large number of unnecessary iteration steps with very small changes in the iterative solutions are performed at the end of the optimization procedure. Even these redundant iteration steps can be interrupted. Indeed, the user can invoke the optimization by the command `OPTIMIZE -f name.dat`. In this case the actual iterative solutions are printed on the screen, and, pushing the `Ctrl C` keys, the user can stop the iteration whenever he observes that the iterative solutions do not improve.

In some cases this method is the fasted local algorithm of `DIPOG-2.1`. For about the same number of iteration steps, the number of function and gradient evaluations is less in the Newton type iteration since multiple function evaluations in the line search are avoided. The price is that, even for accurate gradient computations using high discretization levels, the algorithm may diverge. However, if the initial solution is sufficiently close to a local minimum, then convergence is guaranteed.

10.3.7 Levenberg-Marquardt method

Suppose the objective functional is the least squares sum of deviation terms for efficiencies and phase shifts (cf. (10.2)). Moreover, suppose the optimization problem is to find a local minimum, i.e., to find an admissible vector r_{opt} in $\mathbb{R}^N$ such that $f(r_{opt}) \leq f(r)$ holds at least for any admissible $r \in \mathbb{R}^N$ close to r_{opt}. Here a vector $r \in \mathbb{R}^N$ is called admissible if the coordinates r_i of r satisfy the box constraints $l_i \leq r_i \leq u_i$. The objective funvtional f takes the form

$$f(r) = \|e\|_2^2, \quad e := c - \Phi(r),$$

where Φ maps the admissible parameter sets r into the space $\mathbb{R}^M$ with $M \geq N$, where $c \in \mathbb{R}^M$, and where $\| \cdot \|_2$ is the Euclidean norm. Clearly, the components of $\Phi(r)$ are just the scaled efficiencies or phase shift values for the grating determined by the parameter set r, and the c is the vector of the scaled prescribed values in (10.2). The scaling factors are the square roots of the positive weights in (10.2).

The Levenberg-Marquardt method (ind_opt=6) can easily be applied to find a numerical solution. The method requires no integer input values (ni_opt=0). The number of real parameters is nd_opt=4. These parameters are:

- d_opt[1]: Factor $\mu > 0$, for initial value of regularization parameter for Gauß-Newton equation
- d_opt[2]: Threshold $\varepsilon_1 > 0$, iteration stops if gradient norm satisfies the estimate $\|J^T e\|_\infty < \varepsilon_1$ with J the Jacobian of Φ
- d_opt[3]: Threshold $\varepsilon_2 > 0$, iteration stops if correction Δp of iterative solution p satisfies $\|\Delta p\|_2^2 < \varepsilon_2 \|\Delta p'\|_2^2$ with $\Delta p'$ the value of the previous step
- d_opt[4]: Threshold $\varepsilon_3 > 0$, iteration stops if least square deviation satisfies $\|e\|_2^2 < \varepsilon_3$

To work with default parameters, set d_opt[1]=-1.

10.4 References

For the computation of the gradients of the objective functional, see:

- J. Elschner and G. Schmidt: Diffraction in periodic structures and optimal design of binary gratings I: Direct problems and gradient formulas, *Math. Meth. Appl. Sci.* **21**, pp. 1297–1342 (1998).
- J. Elschner and G. Schmidt: Conical diffraction by periodic structures: Variation of interfaces and gradient formulas, *Math. Nachr.* **252**, pp. 24–42 (2003).

For the numerical optimization methods, see, e.g., the following books and article:

- Ch. Großmann and J. Terno: *Numerik der Optimierung*, Teubner Studienbücher der Mathematik, Teubner Stuttgart, 1997. (augmented Lagrangian method)
- F. Jarre and J. Stoer: *Optimierung*, Springer Verlag New York Berlin Heidelberg, 2004. (interior point method)
- P.J.M. van Laarhoven and E.H.L. Aarts: *Simulated annealing: Theory and*

Applications, D. Reidel Publishing Company, Mathematics and Applications, Member of the Kluwer Academic Publishing Group, Dodrecht Boston Lancaster Tokyo, 1988.
- J. Nocedal and S.J. Wright: *Numerical optimization*, Springer Verlag New-York Berlin Heidelberg, Springer Series in Operation Research, 2000. (conjugate gradient method)
- E.M.Drège, R.M. Al-Assaad, and D.M. Byrne: Mathematical analysis of inverse scatterometry, *Proc.of SPIE* **4689**, pp. 151–162, 2002. (Newton type method)

11 The graphical user interface program `DIPOG-2.1-GUI`

The graphical user interface program `DIPOG-2.1-GUI` can be called e.g. from the directory `GUI`. Using this, all the executables of the directories `CLASSICAL`, `CONICAL`, and `RESULTS` can be invoked. All necessary information is provided by the interface. We emphasize, however, the following.

Via the interface program the user can read a data file "name.dat" (cf. Sects. 5.1 and 7) and change its input data slightly. In any case, a new data file of the same type "name.dat" but with different name will be produced storing the actual input data. Of course, the computational results are written on the screen and into a result file "name.res" (cf. Sect. 5.2).

If the input data is long, then the original programs without graphical user interface are to be preferred since lots of data are easier to handle with data files and editor. Therefore, the input of `DIPOG-2.1-GUI` is restricted. No more than nine upper and lower additional layers are admitted (cf. Sect. 3.2). The grating part without these layers must not contain more than nine different materials. Finally, the input of the geometry by code words (cf. Sect. 3.5) is confined to one line which excludes lamellar, stack, and box gratings as well as profile gratings with parameters. Recall that the interface program `TGUI` allows to construct complex geometries and to include them as "name1.inp" files (cf. Sect. 3.2).

12 Enclosed Files

12.1 Geometry input file “example.inp”

```
#-*-makefile-*-
##################################################################
#                                                                #
#                    #################                           #
#                    #  example.inp  #                           #
#                    #################                           #
#                                                                #
#         all lines beginning with # are comments                #
#                                                                #
##################################################################
#
#      - geometry input file for periodic grating
#      - located in directory ‘‘GEOMETRIES’’
#      - input file for ‘‘gen_polyx’’
#
##################################################################
# Name of the files without extensions ‘‘.inp’’.
#    Output files will have the same name but with
#    tags ‘‘.polyx’’ and ‘‘.sg’’.
# Name:
  example
##################################################################
# Comments.
#    Input must be ended by a ‘‘0’’ in an extra line.
#    These comments will appear in several output and
#    result files.
# Comments:
  This is a fantasy grid
  for the test of gen_polyx!
  0
##################################################################
# Number of materials.
# Must include in its number the two materials of the
# regions immediately above and below the grating structure.
# Number of materials:
  4
##################################################################
# Minimal angle of subdivision triangles:
  20.000000
##################################################################
# Upper bound for mesh size:
  0.500000
##################################################################
# Width of additional strip above and below.
#    Automatic choice of small width
#    if this value is 0.
#    For no additional strip add ‘‘no’’.
#    For additional strip below but no
#    additional strip above, add ‘‘no_up 0.2’’.
#    For additional strip above but no
#    additional strip below, add ‘‘no_lo 0.2’’.
#    (If ‘‘no’’, ‘‘no_up’’ or ‘‘no_lo’’ is added,
#    then the input for the Number of materials
#    must not contain the materials of the
#    excluded strips)
# Width:
  0.200000
#################################################################
# Grid points.
#       #########################################################
#       points of triangulation which is part of the domain
#       for the FEM:
#
#       -> x-components between 0 and 1
#       -> triangles should be disjoint
#       -> union of triangles should be a simply connected domain
#       -> union of triangles should connect the lines x=0,x=1
#       -> union of triangles should be bounded by two vertical
#          lines and by two piecewise linear functions in x
#       -> diameter of each triangle in x-direction must
#          be less than one half (period)
#
#       first add the nodes of all the triangles
#       later give the triangles by the indices of their nodes
#       ########################################################
#    Each point in a separate line.
#    Scaled to period 1.
#    Input ended by ‘‘-1. -1.’’.
# Grid points:
  0.000000 0.800000
  0.500000 0.800000
  0.000000 0.400000
  0.250000 0.400000
  1.000000 0.400000
  0.750000 0.200000
  1.000000 0.200000
  0.000000 0.000000
  0.250000 -0.200000
  1.000000 -0.200000
  0.000000 -0.600000
  1.000000 -0.800000
  -1. -1.
#################################################################
# Triangles.
#    Each given in a separate line by 5 parameters,
#    namely by index of first point, by index of
#    second point, by index of third point, by index
#    of material, and by additional factor for maximal
#    mesh size of partition inside the triangle.
#    Input ended by ‘‘-1 -1 -1 -1 -1.’’.
# Triangles:
  1 3 4 2 1.000000
  4 6 2 2 1.000000
  6 7 5 2 1.000000
  3 8 4 2 1.000000
  8 9 4 2 0.300000
  4 9 6 2 1.000000
  6 10 7 3 1.000000
  8 11 9 3 1.000000
  9 12 6 3 1.000000
  6 12 10 3 1.000000
  -1 -1 -1 -1 -1.

#################################################################
# End
#################################################################
```

12.2 Data file “example.dat” for CLASSICAL

```
#-*-makefile-*-
###################################################################
#                                                                 #
#                     #################                           #
#                     #  example.dat  #                           #
#                     #################                           #
#                                                                 #
#           all lines beginning with # are comments!              #
#                                                                 #
###################################################################
#
#         - input file for ‘‘FEM/GFEM’’
#         - located in directory ‘‘CLASSICAL’’
#
###################################################################
# Name of the output file.
#    The tag ‘‘.res’’ will be added.
#    File will be written into directory ‘‘RESULTS’’.
#    (Alternatively, a path for the location of the file
#     can be added before the name. This must contain at
#     least one slash ’/’. E.g. for a file ‘‘name.res’’
#     in the current working directory write ‘‘  ./name’’)
# Name:
  example
###################################################################
# Should there be an additional output file in the old style of
# DIPOG-1.3 (resp. an eps-file for FEM_CHECK).
#    Add ‘‘  no’’ if not needed.
#    Add ``  yes'' if needed. The name will be the same as
#    the standard output file given above but with the tag
#    ``.erg'' instead of ``.res''.
#    Add ``  phaseshifts'' if no additional output is needed
#    but if phase shifts are preferred instead of Rayleigh
#    coefficients.
# yes or no or phaseshifts:
  yes
###################################################################
# Number of coatings over the grating (N_co_ov).
#    The grating cross section consists of
#    a rectangular area parallel to the axes.
#    This inhomogeneous part is determined
#    by a triangular grid and can have already a few
#    layers of coatings involved. Beneath and above
#    this rectangular structure, there might be additional
#    coated layers of rectangular shape. These kind of
#    layers are called coatings over the grating and
#    coatings beneath the grating, respectively.
#    Alternatively a
#    multilayer system input format is possible:
#    E.g. the input ``MLS n1 n2*n3 n4'' with n1,n2,n3,n4
#    replaced by non-negative integers means
#    N_co_ov=n1+n2*n3+n4 layers with n1 layers above,
#    n2 groups of n3 layers with same widths and materials
#    in the middle, and with 7 layers below.
# Number of coatings:
  2
###################################################################
# Widths of coatings in micro m.
#    Needed only if N_co_ov >0.
#    Else no number no line.
#    For a multilayer system the widths of the
#    n3 layers in the groups must be given only once.
#    I.e. for a multilayer system n1+n3+n4 input numbers
#    are needed.
# Widths:
  0.5
  0.2
###################################################################
# Number of coatings beneath the grating (N_co_be):
  3
###################################################################
# Widths of coatings in micro m.
#    Needed only if N_co_be >0.
#    Else no number no line.
# Widths:
  0.2
  0.3
  0.2
###################################################################
# Wave length in micro m (lambda).
#    Either add a single value e.g. ``.63''.
#    Either add more values by e.g.
#      ‘‘  V
#          5
#          .63
#          .64
#          .65
#          .69
#          .70 ’’.
#      The last means that computation is to be done for
#      the wave lengths from the Vector of length 5:
#      ‘‘.63’’, ‘‘.64’’,‘‘.65’’,‘‘.69’’, and ‘‘.70’’.
#    Or add e.g. ‘‘  I .63 .73 .02’’.
#      The last means that computation is to be done for
#      the wave lengths ‘‘.63+i*.02’’ with i=0,1,2,... and with
#      wave length ‘‘.63+i*.02’’ less or equal to ‘‘.73’’.
#
# Wave length:
  .635
###################################################################
# Temperature in degrees Celsius.
#    From 0 to 400.
#    For room temperature set to 20.
#    Will be ignored for explicitly
#    given refractive indices.
# Temperature:
  20.
###################################################################
# Optical index (refractive index) of cover material.
#    This is c times square root of mu times epsilon.
#    This could be complex like ‘‘4.298 +i 0.073’’ for
#    Si with wave length 500nm.
#    This could be also given by the name of a material
#    like: Air Ag Al Au CsBr Cu InP MgF2 NaCl PMMA PSKL
#          SF5 Si TlBr TlCl Cr ZnS Ge Si1.0 - Si2.0
#          TiO2r Quarz AddOn ... (cf. Userguide)
#    This could be a value interpolated from a user
#    defined table, determined by the name of the file
#    (file is to be located in the current directory,
#    name of file must begin with letter ‘‘u’’ and may
#    consist of no more than five letters like e.g. user,
#    the file consists of lines each with three real
#    numbers, first: wave length in micro meter, second:
#    the real part of the corresponding optical index,
#    third: the imaginary part of the corresponding index).
# Optical index:
  1.0 +i .0
###################################################################
# Optical indices of the materials of the upper coatings.
#    Needed only if N_co_ov >0.
#    Else no number no line.
#    For a multilayer system the indices of the
#    n3 layers in the groups must be given only once.
#    I.e. for a multilayer system n1+n3+n4 input lines
#    are needed.
# Optical indices:
  1.1 +i .0
  1.2 +i .0
###################################################################
# Optical indices of the materials of the lower coatings.
#    Needed only if N_co_be >0.
#    Else no number no line.
# Optical indices:
  2.3 +i .0
  2.2 +i .0
  2.1 +i .0
###################################################################
# Optical index of substrate material.
  2.0 +i .0
###################################################################
# Angle of incident wave in degrees (theta).
#    Either add a single value e.g. ‘‘45.’’.
#    Either add more values by e.g.
#      ‘‘  V
#          5
#          63.
#          64.
#          65.
#          69.
#          70. ’’.
#      The last means that computation is to be done for
#      the angles from the Vector of length 5:
#      ‘‘63.’’, ‘‘64.’’,‘‘65.’’,‘‘69.’’, and ‘‘70.’’.
#    Or add e.g. ‘‘  I 45 56 2’’.
#      The last means that computation is to be done for
#      the angles ‘‘45+i*2’’ with i=0,1,2,... and with
#      angle ‘‘45+i*2’’ less or equal to ‘‘56’’.
#    Note that either the wave length or the angle of
#    incident wave must be single valued.
# Angle of incident wave:
  65
###################################################################
# Type of polarization.
#    Either TE, TM or TE/TM.
# Type:
  TM
###################################################################
# Length factor of additional shift of grating geometry.
#    This is shift into the x-direction, i.e. the
#    direction of the period to the right.
#    This is length of shift relative to period, i.e.
#    the grating structure given by subsequent input
#    will be shifted by factor times the period given
#    in subsequent input.
#    However, only the Rayleigh numbers and efficiencies
#    will be computed according to the shift. The field
#    vectors in the plots are drawn without shift, and
#    the graphics of the executable with tag ‘‘_CHECK’’
#    is drawn without shift!
#    Must be a real number between 0 and 1.
# Length:
  0.
####################################################################
# Stretching factor for grating in y-direction.
#    Must be a positive real number.
# Length:
  1.
####################################################################
# Length of additional shift of grating geometry in micro m.
#    This is shift into the y-direction, i.e. the
#    direction perpendicular to the grating surface
#    pointing into the cover material.
#    Must be a real number.
# Length:
  0.
####################################################################
# Period of grating in micro m:
  1.
####################################################################
# Grating data.
#    Either this should be e.g. ‘‘name1’’ if ‘‘name1.inp’’
#       is the input file with the geometry data in sub-
#       directory ‘‘GEOMETRIES’’.
#       (Alternatively, a path for the location of the file
#        can be added before the name. This must contain at
#        least one slash ’/’. E.g. for a file in the current
#        working directory write ‘‘  ./name1’’)
#    Or this could be a stack of profiles given by the
#       code word stack and many more lines (cf. Userguide)
#    Or this could be
#       e.g. ‘‘  echellea R 0.3 0.03 0.04’’
#       -> ECHELLE GRATING TYPE A (right-angled triangle
#          with hypotenuse parallel to the direction of the
#          periodicity, right interior angle > 45 degrees)
#          with depth of 0.3 micro meter and with coated
#          layers of height 0.03 micro meter resp. 0.04 micro
#          meter over the first resp. second part of the
#          grating (measured in direction perpendicular to
#          echelle profile, height greater or equal to zero)
#       e.g. ‘‘  echellea L 0.3 0.03 0.04’’
#       -> ECHELLE GRATING TYPE A (right-angled triangle
#          with hypotenuse parallel to the direction of the
#          periodicity, left interior angle > 45 degrees)
#          with parameters like above
#       e.g. ‘‘  echellea A 60 0.03 0.04’’
#       -> ECHELLE GRATING TYPE A (right-angled triangle
#          with hypotenuse parallel to the direction of the
#          periodicity) with left interior angle Alpha=60
#          degrees (i.e. depth = period times sin(Alpha)
#          times cos(Alpha)) and other parameters like above
#       e.g. ‘‘  echelleb 60. 0.05’’
#       -> ECHELLE GRATING TYPE B (right-angled triangle
#          with one of the legs parallel to the direction
#          of the periodicity) with angle 60 (angle enclosed
#           by hypotenuse and by the leg parallel to the
#           period) and with a coated layer of height 0.05
#           micro meter (measured in direction perpendicular
#           to echelle profile, height greater or equal to
#           zero)
#        e.g. ``  echelle L 60. R 30. 0.05 0.1''
#        -> GENERAL ECHELLE GRATING with left blaze angle 60
#           degrees,  with right blaze angle 30 degrees, with
#           coated layer over left blaze side of height 0.05
#           micro meter (measured perpendicular to profile,
#           height >= 0) and with coated layer over right blaze
#           side of height 0.1 micro meter (measured
#           perpendicular to profile, height >= 0, must be 0 if
#           previous height is 0). Instead of the two inputs
#           ``L 60.'' and ``R 30.'' one can choose also the
#           inputs ``A 90.'' for an apex angle of 90 degrees or
#           ``D 0.2'' for a depth of grating = 0.2 micro meter.
#           Moreover, any combination of two inputs of the types
#           ``A 90.'', ``L 110.'', ``R 90'', and ``D 0.2'' is
#           accepted. However, the choice ``A 90.'' and ``D 0.2''
#           might be  ambiguous. By definition it fixes an
#           echelle grating with right blaze angle larger than the
#           left. To get the flipped grating with left blaze angle
#           larger than the right, the input should be ``A 90.''
#           and ``D -0.2''.
#        e.g. ``  trapezoid 60. 0.6 3 0.2 0.1 0.1 0.05''
#        -> TRAPEZOIDAL GRATING (trapezoid with the basis
#           parallel to the direction of the periodicity)
#           with angle of 60 degrees (angle enclosed by
#           basis and the sides) with a base of length 0.6
#           micro meter consisting of 3 material layers of
#           heights 0.2, 0.1, and 0.1 micro meter,
#           respectively, and with a coated layer of height
#           0.05 micro meter (greater or equal to zero)
#        e.g. ``  mtrapezoid
#                 3
#                 0.005 0.015 0.005
#                 80.   90.   80.
#                 0.01
#                 7
#                 0.05 0.075 0.05 0.05 0.075 0.075 0.05
#                 0.15 0.3   0.4  0.5  0.6   0.75  0.9  ''
#        -> MULTITRAPEZOIDAL GRATING (finite number of symmetric
#           trapezoidal bridges located side by side)
#           each bridge consists of 3 trapezoidal layers, one over
#           each other; hights in micro meter of these layers from
#           above to below are 0.005, 0.015, and 0.005; sidewall
#           angles in degrees of these layers from above to below
#           are 80, 90, 80; height in micro meter at which the
#           lateral width of the bridge is given is 0.01; number of
#           trapezoidal bridges per period is 7; lateral widths in
#           micro meter of these bridges are 0.05, 0.075, 0.05, 0.05,
#           0.075, 0.075, and 0.05; x-coordinates in micro meter of
#           the mid-points of the bottom lines of these bridges are
#          0.15, 0.3, 0.4, 0.5, 0.6, 0.75, and 0.9.
#       e.g. ``  lAmellar 3 4
#                0.2 0.6
#               -0.2 1.0
#                0.   0.5   0.7
#                0.0  0.50  0.90
#                0.00 0.500 0.900  ''
#       -> LAMELLAR GRATING (rectangular grating consisting
#          of several materials placed in rectangular sub-
#          domains) with 3 columns each divided into 4
#          rectangular layers, first column with x coordinate
#          in 0<x<0.2 (given in micro meter), second column with 0.2<x<0.6,
#          third columnn with 0.6<x<period (period given above),
#          whole grating with y coordinate s.t. -0.2<y<1.0,
#          first column: first layer with -0.2<y<0., second with
#          0.<y<0.5, third with 0.5<y<0.7 and fourth with 0.7<y<1,
#          second column: first layer with -0.2<y<0.0, second with
#          0.0<y<0.50, third with 0.50<y<0.90 and fourth with 0.90<y<1.,
#          third column: first layer with -0.2<y<0.00, second with
#          0.00<y<0.500, third with 0.500<y<0.900 and fourth with
#          0.900<y<1.
#       e.g. ``  lAmellar 1 1
#                0.2 0.8     ''
#       -> SIMPLE LAYER (special case of lamellar grating) with
#          layer material s.t. y-coordinate satisfies 0.2<y<0.8
#          (given in micro meter).
#       e.g. ``  polygon file1''
#       -> GRATING DETERMINED BY A POLYGONAL LINE defined
#          by the data in the file with name ``../GEOMETRIES/file1''
#          (in ``../GEOMETRIES/file1'': in each line beginning without `#'
#          there should be the x- and y-coordinate of one
#          of the consecutive corner points, first point
#          with x-coordinate 0, last point with x-coordinate
#          1, same y-coordinate for first and last point,
#          all x-coordinates between 0 and 1, at least two
#          different y-coordinates, last line should be
#          ``End'')
#       e.g. ``  polygon2 file1 file2''
#       -> COATED GRATING DETERMINED BY POLYGONAL LINES,
#          i.e. grating profile line is defined by the data
#          in the file with name ``../GEOMETRIES/file1''
#          (in ``../GEOMETRIES/file1'':
#          in each line beginning without `#' there should
#          be the x- and y-coordinate of one of the
#          consecutive corner points, first point with
#          x-coordinate 0, last point with x-coordinate
#          1, same y-coordinate for first and last point,
#          all x-coordinates between 0 and 1, at least two
#          different y-coordinates, last line should be
#          ``End'') and the coated layer is enclosed between
#          the polygonal line of ``../GEOMETRIES/file1''
#          and the polygonal line of the file with name
#          ``../GEOMETRIES/file2''
#          (in ``../GEOMETRIES/file2'':
```

```
#           in each line beginning without ‘#’ there should be
#           the x- and y-coordinate of one of the consecutive
#           corner points, first and last point must be corner
#           of first polygon, second polygon must be on left-
#           hand side of first, one to one correspondence of
#           the corners on the two polygons between first and
#           last point of second polygon, quadrilateral between
#           corresponding segments on the left of first
#           polygon, these quadrilaterals must be disjoint, last
#           line should be ‘‘End’’)
#        e.g. ‘‘  profile’’
#        -> GRATING DETERMINED BY A SMOOTH PARAMETRIC CURVE,
#           i.e. grating determined by profile line given as
#           {(fctx(t),fcty(t)):0<=t<=1}, where the functions
#           ‘‘t|->fctx(t)’’ and ‘‘t|->fcty(t)’’ are defined
#           by the ‘‘c’’-code of the file ‘‘../GEOMETRIES/profile.c’’
#        e.g. ‘‘  profile_par 2 3
#                 1
#                 0
#                 1.5
#                 0.2
#                 0.3              ’’
#        -> GRATING DETERMINED BY A SMOOTH PARAMETRIC CURVE,
#           WITH PARAMETERS,
#           i.e. grating determined by profile line given as
#           {(fctx(t),fcty(t)):0<=t<=1}, where the functions
#           ‘‘t|->fctx(t)’’ and ‘‘t|->fcty(t)’’ are defined
#           by the ‘‘c’’-code of ‘‘../GEOMETRIES/profile_par.c’’;
#           the last code uses 2 integer parameters and 3
#           real parameters named IPARaM1, IPARaM2, RPARaM1,
#           RPARaM2, RPARaM3;
#           the integer parameters take the values 1 and 0
#           following the first line of the calling sequence
#           and the real parameters take the values 1.5, 0.2,
#           and 0.3 following the integer parameter values
#           (Any number of parameters is possible for a
#           corresponding file ‘‘../GEOMETRIES/profile_par.c’’.)
#        e.g. ‘‘  profile 0.125*sin(2.*M_PI*t)’’
#        -> GRATING DETERMINED BY A SIMPLE SMOOTH FUNCTION,
#           i.e. grating determined by sine profile line given
#           as {(t,fcty(t)):0<=t<=1}, where the function
#           ‘‘t|->fcty(t)’’ is defined by the ‘‘c’’-code
#           fcty(t)=0.125*sin(2.*M_PI*t),
#           (do not use any ‘‘blank’’/‘‘space’’ in the c-code)
#        e.g. ‘‘  profile 0.5+0.5*cos(M_PI*(1.-t)) 0.25*sin(M_PI*t)’’
#        -> GRATING DETERMINED BY A SIMPLE SMOOTH PARAMETRIC CURVE,
#           i.e., grating determined by ellipsoidal profile line
#           given as {(fctx(t),fcty(t)):0<=t<=1}, where the
#           functions ‘‘t|->fcty(t)’’ and ‘‘t|->fcty(t)’’ are
#           defined by the ‘‘c’’-codes
#           fctx(t)=0.5+0.5*cos(M_PI*(1.-t)) and
#           fcty(t)=0.25*sin(M_PI*t), respectively
#           (do not use any ‘‘blank’’/‘‘space’’ in the c-codes)
#        e.g. ‘‘  profiles’’
#        -> GRATING DETERMINED BY SMOOTH PARAMETRIC CURVES,
#           i.e. grating determined by profile lines given as
#           {(fctx(j,t),fcty(j,t)):0<=t<=1}, j=1,...,n=nmb_curves,
#           where the functions ``t|->fctx(j,t)'' and
#           ``t|->fcty(j,t)'' are defined by the ``c''-code
#           of the file ``../GEOMETRIES/profiles.c''
#        e.g. ``  profiles_par 1 2
#                 3
#                 0.5
#                 0.50            ''
#        -> GRATING DETERMINED BY SMOOTH PARAMETRIC CURVES,
#           WITH PARAMETERS,
#           i.e. grating determined by profile lines given as
#           {(fctx(j,t),fcty(j,t)):0<=t<=1}, j=1,...,n=nmb_curves,
#           where the functions ``t|->fctx(j,t)'' and
#           ``t|->fcty(j,t)'' are defined by the ``c''-code
#           of the file ``../GEOMETRIES/profiles_par.c'';
#           the last code uses 1 integer parameter and 2
#           real parameters named IPARaM1, RPARaM1, RPARaM2;
#           the integer parameter takes the value 3
#           following the first line of the calling sequence
#           and the real parameters take the values 0.5 and 0.50
#           following the integer parameter values
#           (Any number of parameters is possible for a
#           corresponding file ``../GEOMETRIES/profiles_par.c''.)
#        e.g. ``  pin''
#        -> PIN GRATING DETERMINED BY PARAMETRIC CURVE,
#           i.e. over a flat grating with surface {(x,0):0<=x<=1}
#           a material part is attached which is located between
#           {(x,0):0<=x<=1} and {(fctx(t),fcty(t)):0<=t<=1}.
#           Here {(fctx(t),fcty(t)):0<=t<=1} is a simple open
#           arc connecting (fctx(0),fcty(0))=(xmin,0) with
#           (fctx(1),fcty(1))=(1-xmin,0) such that 0<xmin<0.5
#           is a fixed number, such that 0<fctx(t)<1, 0<t<1, and
#           such that 0<fcty(t), 0<t<1. The functions fctx, fcty
#           and the parameter xmin are defined by the code in
#           ``../GEOMETRIES/pin.c''.
#        e.g. ``  cpin''
#        -> COATED PIN GRATING DETERMINED BY TWO PARAMETRIC CURVES,
#           i.e. over a flat grating with surface {(x,0):0<=x<=1}
#           a material part is attached which is located between
#           {(x,0):0<=x<=1} and {(fctx(1,t),fcty(1,t)):0<=t<=1}.
#           Here {(fctx(1,t),fcty(1,t)):0<=t<=1} is a simple open
#           arc connecting (fctx(1,0),fcty(1,0))=(xmin,0) with
#           (fctx(1,1),fcty(1,1))=(1-xmin,0) such that 0<xmin<0.5
#           is a fixed number, such that 0<fctx(1,t)<1, 0<t<1, and
#           such that 0<fcty(1,t), 0<t<1. Additionaly, a coating
#           layer is attached located between the first curve
#           {(fctx(1,t),fcty(1,t)):0<=t<=1} and a second curve
#           {(fctx(2,t),fcty(2,t)):0<=t<=1}. The last connects the
#           point (fctx(1,arg1),fcty(1,arg1))=(fctx(2,0),fcty(2,0))
#           with (fctx(1,arg2),fcty(1,arg2))=(fctx(2,1),fcty(2,1)).
#           Moreover, {(fctx(2,t),fcty(2,t)):0<=t<=1} is a simple open
#           arc above {(fctx(1,t),fcty(1,t)):0<=t<=1} such that
#          0<fctx(2,t)<1, 0<t<1. The functions fctx(1,.), fctx(2,.),
#          fcty(1,.), and fcty(2,.) and the parameters arg1, arg2,
#          xmin are defined by the code of the file
#          ``../GEOMETRIES/cpin.c''.
#       e.g. ``  cpin2''
#       -> COATED PIN GRATING DETERMINED BY TWO PARAMETRIC CURVES
#          TYPE 2,
#          i.e. over a flat grating with surface {(x,0):0<=x<=1}
#          a material part is attached which is located between
#          {(x,0):0<=x<=1} and {(fctx(1,t),fcty(1,t)):0<=t<=1}.
#          Here {(fctx(1,t),fcty(1,t)):0<=t<=1} is a simple open
#          arc connecting (fctx(1,0),fcty(1,0))=(xmin,0) with
#          (fctx(1,1),fcty(1,1))=(1-xmin,0) such that 0<xmin<0.5
#          is a fixed number, such that 0<fctx(1,t)<1, 0<t<1, and
#          such that 0<fcty(1,t), 0<t<1. Additionaly, a coating
#          layer is attached located between the first curve
#          {(fctx(1,t),fcty(1,t)):0<=t<=1} and a second curve
#          {(fctx(2,t),fcty(2,t)):0<=t<=1}. The last connects the
#          point (x1,0)=(fctx(2,0),fcty(2,0)) with (x2,0)=
#          (fctx(2,1),fcty(2,1)) with 0<x1<xmin<1-xmin<x2. Moreover,
#          {(fctx(2,t),fcty(2,t)):0<=t<=1} is a simple open
#          arc above {(fctx(1,t),fcty(1,t)):0<=t<=1} such that
#          0<fctx(2,t)<1, 0<t<1. The functions fctx(1,.), fctx(2,.),
#          fcty(1,.), and fcty(2,.) and the parameter xmin are
#          defined by the code of the file ``../GEOMETRIES/cpin2.c''.
#       e.g. ``  stack 3
#                profile t /##/ 0.2*sin(2.*M_PI*t)
#                2.
#                profile 0.2*sin(2.*M_PI*t)
#                1.
#                profile t /##/ 0.
#                0.                  ''
#       -> STACK GRATING,
#          i.e. a stack of 3 profile curves shifted by 2, 1, 0
#          micro meter in vertical direction. For more details, see
#          the description in the USERGUIDE.
#       e.g. ``  bOX 0.1
#                -1. 1.
#                2 3
#                { 2.*t }
#                { -0.6 }
#                { 1.+0.4*cos(2.*M_PI*t)}
#                { 0.4*sin(2.*M_PI*t) }
#                1. 0.8
#                1. 0.
#                1. -0.8 ''
#       -> BOX GRATING,
#          i.e. a box geometry of size [0.,period=2.]x[-1.,1.]
#          given in micro m (period is supposed to be set to
#          2. in previous input lines); number 0.1 behind word
#          BOX is a mesh size factor; [0.,2.]x[-1.,1.] is
#          divided by 2 curves into 3 different material parts;
#          curves are given by the c-code in following 2 times
#          two lines; first code line is x-component of first
#          curve; second code is y-component of first curve;
#          third code is x-component of second curve; etc.;
#          last three input lines define material points;
#          material with index one is located in part of box
#          separated by above curves and containing (1.,0.8);
#          material with index two is in part of box separated
#          by above curves and containing (1.,0.); etc.; Note
#          that the curves must be simple not self intersecting,
#          contained in the box; allowed intersection points
#          of two different curves are end points only; material
#          areas separated by curves must be such that area with
#          first index contains strip beneath upper boundary side
#          of box and that area with last index contains strip
#          above lower boundary side.
#       e.g. ``  rough_mls name ''
#       -> MULTILAYER SYSTEM WITH ROUGH INTERFACES,
#          i.e. nla layers + nmld times nld layers + nlb layers,
#          This system is described by the file ``name'', which
#          is contained in the directory ``../GEOMETRIES''.
#          (alternatively, the name must contain the path of the file)
#          This input file ``name'' contains the following ordered
#          data each in a separate line (comment lines begin with '#'):
#            - dummy file name
#            - width of additional layer above and below the structure
#              (must be positive or could be 'no' for no additional layer)
#            - period of grating 'per'
#            - number 'nla' of layers above multilayer system
#            - number 'nlb' of layers below multilayer system
#            - number 'nmld' of multilayers inbetween
#            - number 'nld' of layers in each multilayer
#            - number 'ncorn' of corner points in each polygonal approximation
#              (interfaces=randomly generated polygon)
#            - number 'nrand' of standard Gaussian distributed random numbers
#              needed to generate interfaces
#              (automatically created if nrand=0, otherwise 'nrand' has
#               to equal ncorn*(nla+nld*nmld+nlb))
#            - (optional) standard Gaussian distributed random numbers
#            - 'nla' widths of layers above multilayer system,
#            - 'nld' widths of layers in each multilayer,
#            - 'nlb' widths of layers below multilayer system,
#            - standard deviation 'sigma[i]' of each layer interface,
#              for i=0,...,('nla'+'nld'*'nmld'+'nlb'-1)
#            - correlation length 'corl[i]' of each layer interface,
#              for i=0,...,('nla'+'nld'*'nmld'+'nlb'-1)
#          Note that the first of the subsequent refractive indices of
#          the grating must be that of the superstrate resp. that of
#          the adjacent upper layer. If the width of the additional
#          layers is positive, then the last of the subsequent refractive
#          indices of the grating must be that of the substrate resp. that
#          of the adjacent lower layer.
#       e.g. ``  rough_mls k name ''
#       -> MULTILAYER SYSTEM WITH ROUGH INTERFACES k TIMES,
#          just like above. However, k random realizations are computed
#          and the output values are replaced by the mean values over
#          the k computations. Standard deviations are added, too.
# Grating data:
  example
###################################################################
# Number of different grating materials (N_mat).
#    This includes the material of substrate and cover material.
#    For example,
#       if Grating data is  ‘‘  name1’’
#       -> Number of materials given in file ‘‘name1.inp’’
#       if Grating data is  ‘‘  echellea ...’’
#       -> N_mat = 3 with coating height >0
#          N_mat = 2 with coating height =0
#       if Grating data is  ‘‘  echelleb ...’’
#       -> N_mat = 3 with coating height >0
#          N_mat = 2 with coating height =0
#       if Grating data is  ‘‘  echelle ...’’
#       -> N_mat = 3 with coating heights >0
#          N_mat = 2 with coating heights =0
#       if Grating data is  ‘‘  trapezoid ...’’
#       -> N_mat = number of material layers +3
#                  for coating height >0
#          N_mat = number of material layers +2
#                  for coating height =0
#       if Grating data is  ‘‘  mtrapezoid ...’’
#       -> N_mat = number of material layers in each bridge +2
#       if Grating data is  ‘‘  lAmellar k m
#                            ...  ’’
#       -> N_mat = k times m plus 2
#       if Grating data is  ‘‘  polygon file1’’
#       -> N_mat = 2
#       if Grating data is  ‘‘  polygon2 file1 file2’’
#       -> N_mat = 3
#       if Grating data is  ‘‘  profile’’
#       -> N_mat = 2
#       if Grating data is  ‘‘  profile 2 3
#                            ...        ’’
#       -> N_mat = 2
#       if Grating data is  ‘‘  profile 0.125*sin(2.*M_PI*t)’’
#       -> N_mat = 2
#       if Grating data is
#       ‘‘  profile 0.5+0.5*cos(M_PI*(1.-t)) 0.25*sin(M_PI*t)’’
#       -> N_mat = 2
#       if Grating data is ‘‘  profiles’’
#       -> N_mat = n+1 with n=nmb_curves from
#                  the file ‘‘../GEOMETRIES/profiles.c’’
#       if Grating data is ‘‘  profiles ... ’’
#       -> N_mat = n+1 with n=nmb_curves from
#                  the file ‘‘../GEOMETRIES/profiles_par.c’’
#       if Grating data is ‘‘  pin ’’
#       -> N_mat = 3
#       if Grating data is ‘‘  cpin ’’
#       -> N_mat = 4
#       if Grating data is ‘‘  cpin2 ’’
#       -> N_mat = 4
#       if Grating data is ‘‘  stack k’’
#       -> N_mat = k+1
#       if Grating data is ‘‘  bOX ...’’
#       -> N_mat = number indicated as second
#                  integer in second lind after
#                  line with code word bOX
#       if Grating data is ‘‘  rough_mls ...’’
#       -> N_mat = nla+nld+nlb+2 if width of additional
#                  layer above and below is positive
#          N_mat = nla+nld+nlb+1 if instead of the value for
#                  the width of additional layer above and below
#                  a ’no’ is given in the input file
# Number of materials:
  4
####################################################################
# Optical indices of grating materials.
#    This is c times square root of mu times epsilon.
#    If meaningful, then the refractive indices should be ordered
#    according to the location from above to below.
#    If an input file ‘‘name1.inp’’ is used, then the optical index
#    of a subdomain with the material index j is just the j-th
#    optical index following below.
#    If grating is ‘‘lAmellar ...’’, then first material is cover
#    material, last material is substrate, and all other materials
#    are ordered from left to right and inside the columns from
#    below to above.
#    For technical reasons, the index of the material adjacent to
#    the upper line of the grating structure must coincide with
#    that of the material in the adjacent upper coated layer resp.
#    in the adjacent superstrate. Similarly the index of the
#    materials adjacent to the lower line of the grating structure
#    must coincide with that of the material in the adjacent lower
#    coated layer resp. in the adjacent substrate.
#    N_mat numbers are needed.
# Optical indices:
  1.2 +i .0
  1.5 +i .0
  1.7 +i .0
  2.3 +i .0
####################################################################
# Number of levels (Lev).
#    In each refinement step the step size of the mesh is halved.
#    Number of refinement steps is Lev.
#    (Alternatively, one can prescribe an bound for the maximal
#     error of the efficiencies. E.g. the input ‘‘e 1.’’ means
#     that the level for the computation is the smallest positive
#     integer such that all efficiencies are computed with an
#     estimated error less than 1 per cent.)
# Number:
  3
####################################################################
# End
####################################################################
```

12.3 Data file "generalized.Dat" for CLASSICAL resp. "conical.Dat" in CONICAL

```
#-*-makefile-*-
############################################################
#                                                          #
#                #####################                     #
#                #  generalized.Dat  #                     #
#                #####################                     #
#                                                          #
#        all lines beginning with # are comments           #
#                                                          #
############################################################
#
#         - input file for ‘‘GFEM’’
#         - located in directory ‘‘CLASSICAL’’
#         - contains constants for numerical
#           method in program ‘‘GFEM’’
#
############################################################
#
#  Recommendation for n_DOF and n_LFEM:
#  ------------------------------------
#
#       a) mild accuracy requirements
#          and wave numbers not too large:
#
#          n_DOF=1,3,7  n_LFEM=2*n_DOF+1
#
#       b) challenging accuracy requirements
#          or large wave numbers:
#
#          n_DOF=3  with  n_LFEM=31   or
#          n_DOF=7  with  n_LFEM=127  or
#          n_DOF=15 with  n_LFEM=511
#
#  Recommendation for n_UPA:
#  -------------------------
#
#       Take the first level l_0 (cf. the last input in
#       ‘‘name.dat’’ which is the upper bound for all
#       levels to be computed, and cf. the levels
#       indicated in the result files ‘‘name2.res’’)
#       such that the next level results in about four
#       times the number of grid points. Then, if you
#       wish to compute on level l_0+l_1, set the
#       maximum level of computation (last input in
#       ‘‘name.dat’’) to l_0 and choose n_UPA as 2 to
#       the power l_1.
#
############################################################
# n_DOF.
#    Additional degrees of freedom on each triangle side.
#    Indeed, trial functions on each subdivision triangle
#    are approximate solutions of pde s.t. restriction to
#    triangle sides coincides with Lagrange interpolation
#    polynomials on triangle side (Dirichlet’s problem).
#    Here interpolation is taken over uniform grid with
#    [n_DOF+2] interpolation knots.
#    Value should satisfy 0<=n_DOF<100.
# value:
  3
############################################################
# n_LFEM.
#    Approximate solution determined by FEM over subdivision
#    triangle, where additional uniform FEM partition on
#    each small triangle is chosen such that the step size
#    is side length divided by [n_LFEM+1].
#    If n_LFEM=1, n_DOF=0: conventional FEM method.
#    If n_DOF=n_LFEM: conventional FEM method with
#                     elimination of interior nodes of grid
#                     triangle, i.e. real mesh size is mesh
#                     size shown in result file divided by
#                     [n_DOF+1].
#    If n_DOF<n_LFEM: method resembles p=method or PUM.
#    Value should satisfy 1<=n_LFEM<2048 and
#    [n_LFEM+1] must be a multiple of [n_DOF+1].
# value:
  63
############################################################
# n_UPA.
#    This is for additional uniform partition of all primary
#    grid triangles into n_UPA*n_UPA equal subdomains,
#    i.e. original side of grid triangle is split
#    into n_UPA sides of uniform partition subtriangles.
#    Value should satisfy 1<=n_UPA<=128.
# value:
  1
############################################################
#
# that’s it
#
############################################################
```

12.4 Data file “example.dat” for CONICAL

```
#-*-makefile-*-
######################################################################
#                                                                    #
#                     #################                              #
#                     #  example.dat  #                              #
#                     #################                              #
#                                                                    #
#         all lines beginning with ‘‘#’’ are comments!               #
#                                                                    #
######################################################################
#
#         - input file for ‘‘FEM/GFEM’’
#         - located in directory ‘‘CONICAL’’
#
######################################################################
# Name of the output file.
#    The tag ‘‘.res’’ will be added.
#    The file will be written in the ‘‘RESULTS’’ directory.
#    (Alternatively, a path for the location of the file
#     can be added before the name. This must contain at
#     least one slash ’/’. E.g. for a file ‘‘name.res’’
#     in the current working directory write ‘‘  ./name’’)
# Name:
  example
###################################################################
# Should there be an additional output file in the old style of
# DIPOG-1.3 (resp. an eps-file for FEM_CHECK).
#    Add ‘‘  no’’ if not needed.
#    Add ‘‘  yes’’ if needed. The name will be the same as
#    the standard output file given above but with the tag
#    ‘‘.erg’’ instead of ‘‘.res’’.
# yes or no:
  yes
######################################################################
# Number of coating layers over the grating (N_co_ov).
#    The grating cross section consists of a rectangular area
#    parallel to the axes. This inhomogeneous part is determined
#    by a triangular grid and can have already a few
#    layers of coatings involved. Beneath and above
#    this rectangular structure, there might be additional
#    coated layers of rectangular shape. These kind of
#    layers are called coating layers over the grating and
#    coating layers beneath the grating, respectively.
#    Alternatively a
#    multilayer system input format is possible:
#    E.g. the input ‘‘MLS n1 n2*n3 n4’’ with n1,n2,n3,n4
#    replaced by non-negative integers means
#    N_co_ov=n1+n2*n3+n4 layers with n1 layers above,
#    n2 groups of n3 layers with same widths and materials
#    in the middle, and with 7 layers below.
# Number:
  2
######################################################################
# Widths of coating layers in micro meter.
#    N_co_ov entries. Needed only if N_co_ov >0.
#    Else no entry and no line.
#    For a multilayer system the widths of the
#    n3 layers in the groups must be given only once.
#    I.e. for a multilayer system n1+n3+n4 input numbers
#    are needed.
# Widths:
  0.05
  0.03
######################################################################
# Number of coating layers beneath the grating (N_co_be):
  1
######################################################################
# Widths of coating layers in micro meter.
#    N_co_be entries. Needed only if N_co_be >0.
#    Else no entry and no line.
# Widths:
  0.05
######################################################################
# Wave length in micro meter (lambda).
#    Either add a single value e.g. ``.63''.
#    Either add more values by e.g.
#      ``  V
#          5
#          .63
#          .64
#          .65
#          .69
#          .70  ''.
#      The last means that computation is to be done for
#      the wave lengths from the Vector of length 5:
#      ``.63'', ``.64'',``.65'',``.69'', and ``.70''.
#    Or add e.g.
#      ``  I .63 .73 .02''.
#      The last means that computation is to be done for
#      the wave lengths ``.63+i*.02'' with i=0,1,2,... and with
#      wave length ``.63+i*.02'' less or equal to ``.73''.
# Wave length:
  .635
######################################################################
# Temperature in degrees Celsius from 0 to 400.
#    20. for room temperature!
#    Must be set to any fixed number.
#    Will be ignored if optical indices are given explicitly.
# Temperature:
  20.
######################################################################
# Optical index (refractive index) of cover material.
#    This is c times square root of mu times epsilon.
#    This could be complex like e.g. ``4.298 +i 0.073'' for
#    Si with wave length 500nm.
#    This could be also given by the name of a material
#    like: Air Ag Al Au CsBr Cu InP MgF2 NaCl PMMA PSKL
#          SF5 Si TlBr TlCl Cr ZnS Ge Si1.0 - Si2.0
#          TiO2r Quarz AddOn ... (cf. Userguide)
#    This could be a value interpolated from a user
#    defined table, determined by the name of the file
#    (file is to be located in the current directory,
#    name of file must begin with letter ``u'' and may
#    consist of no more than five letters like e.g. user,
#    the file consists of lines each with three real
#    numbers, first: wave length in micro meter, second:
#    the real part of the corresponding optical index,
#    third: the imaginary part of the corresponding index).
# Optical index:
  Air
######################################################################
# Optical indices of the materials of the upper coating layers.
#    This is c times square root of mu times epsilon.
#    N_co_ov entries. Needed only if N_co_ov >0.
#    Else no entry and no line.
#    For a multilayer system the indices of the
#    n3 layers in the groups must be given only once.
#    I.e. for a multilayer system n1+n3+n4 input lines
#    are needed.
# Optical indices:
  1.2
  1.3
######################################################################
# Optical indices of the materials of the lower coating layers.
#    This is c times square root of mu times epsilon.
#    N_co_be entries. Needed only if N_co_be >0.
#    Else no entry and no line.
# Optical indices:
  1.6
######################################################################
# Optical index of substrate material.
#    This is c times square root of mu times epsilon.
# Optical index:
  1.5 +i 0.
######################################################################
# Type of output results.
#    Either ``TE/TM'': results in terms of TE and TM part of Wave.
#    Either ``Jones'': results in terms of Jones vector
#                      representation.
#    Or ``3.Com'':     results in terms of the component in the z
#                      axis, that is in the direction of the grooves.
#    For more details cf. Section 2.3 in USERGUIDE.ps.
# Type:
  3.Com
######################################################################
# Type of polarization and coordinate system for incoming wave vector.
#    Either ``TE'': means that incident electric field is perpendicular
#                   to wave vector and to normal of grating plane
#                   (plane of grating grooves) and
#                   incoming wave vector is presented in xz system as
#                   (sin theta cos phi, -cos theta, sin theta sin phi).
#    Either ‘‘TM’’: means that incident magnetic field is perpendicular
#                   to wave vector and to normal of grating plane and
#                   incoming wave vector is presented in xz system as
#                   (sin theta cos phi, -cos theta, sin theta sin phi).
#    Either ‘‘TE/TM’’:
#                   means TE and TM, two calculations.
#    Either ‘‘TP’’: means polarized electro-magnetic field and
#                   incoming wave vector is presented in xz system as
#                   (sin theta cos phi, -cos theta, sin theta sin phi).
#    Or ‘‘pol’’:    means polarized electro-magnetic field and
#                   incoming wave vector is presented in xy system as
#                   (sin theta cos phi, -cos theta cos phi, sin phi).
# Type:
  TE/TM
######################################################################
# Parameter of polarization.
#    If type of polarization is ‘‘pol’’ or ‘‘TP’’, then this is
#    the angle (in degrees) between x axis (axis in plane of
#    grating grooves which is perpendicular to grooves)
#    and projection of electric field vector onto x-z plane of
#    grating grooves.
#    Needed only if polarization is of type ‘‘pol’’ or ‘‘TP’’.
#    Else no entry and no line.
# Parameter:
######################################################################
# Angle of incident wave in degrees (theta).
#    If type of polarization is ‘‘pol’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta cos phi, sin phi)
#    with the restriction -90 < phi,theta < 90.
#    If type of polarization is ‘‘TE’’/‘‘TM’’/‘‘TP’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta, sin theta sin phi)
#    with the restriction 0 < theta < 90.
#    Either add a single value e.g. ‘‘45.’’.
#    Either add more values by e.g.
#      ‘‘  V
#          5
#          63.
#          64.
#          65.
#          69.
#          70. ’’.
#      The last means that computation is to be done for
#      the angles from the Vector of length 5:
#      ‘‘63.’’, ‘‘64.’’,‘‘65.’’,‘‘69.’’, and ‘‘70.’’.
#    Or add e.g.
#      ‘‘  I 45 56 2’’.
#      The last means that computation is to be done for
#      the angles ‘‘45+i*2’’ with i=0,1,2,... and with
#      angle ‘‘45+i*2’’ less or equal to ‘‘56’’.
#    Note that either the wave length or the angle of
#    incident wave theta must be single valued.
# Angle:
  30.
######################################################################
# Angle of incident wave in degrees (phi).
#    If type of polarization is ‘‘pol’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta cos phi, sin phi)
#    with the restriction -90 < phi,theta < 90.
#    If type of polarization is ‘‘TE’’/‘‘TM’’/‘‘TP’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta, sin theta sin phi)
#    with the restriction 0 < theta < 90.
#    Either add a single value e.g. ‘‘45.’’.
#    Either add more values by e.g.
#      ‘‘  V
#          5
#          63.
#          64.
#          65.
#          69.
#          70. ’’.
#      The last means that computation is to be done for
#      the angles from the Vector of length 5:
#      ‘‘63.’’, ‘‘64.’’,‘‘65.’’,‘‘69.’’, and ‘‘70.’’.
#    Or add e.g.
#      ‘‘  I 45 56 2’’.
#      The last means that computation is to be done for
#      the angles ‘‘45+i*2’’ with i=0,1,2,... and with
#      angle ‘‘45+i*2’’ less or equal to ‘‘56’’.
#    Note that two of the three, the wave length, the
#    angle of incident wave theta, and the angle of
#    incident wave phi, must be single valued.
# Angle:
  47.
###################################################################
# Length factor of additional shift of grating geometry.
#    This is shift into the x-direction, i.e. the
#    direction of the period to the right.
#    This is length of shift relative to period, i.e.
#    the grating structure given by subsequent input
#    will be shifted by factor times the period given
#    in subsequent input.
#    However, only the Rayleigh numbers and efficiencies
#    will be computed according to the shift. The field
#    vectors in the plots are drawn without shift, and
#    the graphics of the executable with tag ‘‘_CHECK’’
#    is drawn without shift!
#    Must be a real number between 0 and 1.
# Length:
  0.
###################################################################
# Stretching factor for grating in y-direction.
#    Must be a positive real number.
# Length:
  1.
###################################################################
# Length of additional shift of grating geometry in micro m.
#    This is shift into the y-direction, i.e. the
#    direction perpendicular to the grating surface
#    pointing into the cover material.
#    Must be a real number.
# Length:
  0.
######################################################################
# Period of grating in micro meter:
  1.
###################################################################
# Grating data.
#    Either this should be e.g. ‘‘name1’’ if ‘‘name1.inp’’
#       is the input file with the geometry data in sub-
#       directory ‘‘GEOMETRIES’’.
#       (Alternatively, a path for the location of the file
#        can be added before the name. This must contain at
#        least one slash ’/’. E.g. for a file in the current
#        working directory write ‘‘  ./name1’’)
#    Or this could be a stack of profiles given by the
#       code word stack and many more lines (cf. Userguide)
#    Or this could be
#       e.g. ‘‘  echellea R 0.3 0.03 0.04’’
#       -> ECHELLE GRATING TYPE A (right-angled triangle
#          with hypotenuse parallel to the direction of the
#          periodicity, right interior angle > 45 degrees)
#          with depth of 0.3 micro meter and with coated
#          layers of height 0.03 micro meter resp. 0.04 micro
#          meter over the first resp. second part of the
#          grating (measured in direction perpendicular to
#          echelle profile, height greater or equal to zero)
#       e.g. ‘‘  echellea L 0.3 0.03 0.04’’
#       -> ECHELLE GRATING TYPE A (right-angled triangle
#          with hypotenuse parallel to the direction of the
#          periodicity, left interior angle > 45 degrees)
#          with parameters like above
#       e.g. ‘‘  echellea A 60 0.03 0.04’’
#       -> ECHELLE GRATING TYPE A (right-angled triangle
#          with hypotenuse parallel to the direction of the
#          periodicity) with left interior angle Alpha=60
#          degrees (i.e. depth = period times sin(Alpha)
#          times cos(Alpha)) and other parameters like above
#       e.g. ‘‘  echelleb 60. 0.05’’
#       -> ECHELLE GRATING TYPE B (right-angled triangle
#          with one of the legs parallel to the direction
#          of the periodicity) with angle 60 (angle enclosed
#          by hypotenuse and by the leg parallel to the
#          period) and with a coated layer of height 0.05
#          micro meter (measured in direction perpendicular
#          to echelle profile, height greater or equal to
#            zero)
#         e.g. ``  echelle L 60. R 30. 0.05 0.1''
#         -> GENERAL ECHELLE GRATING with left blaze angle 60
#            degrees,  with right blaze angle 30 degrees, with
#            coated layer over left blaze side of height 0.05
#            micro meter (measured perpendicular to profile,
#            height >= 0) and with coated layer over right blaze
#            side of height 0.1 micro meter (measured
#            perpendicular to profile, height >= 0, must be 0 if
#            previous height is 0). Instead of the two inputs
#            ``L 60.'' and ``R 30.'' one can choose also the
#            inputs ``A 90.'' for an apex angle of 90 degrees or
#            ``D 0.2'' for a depth of grating = 0.2 micro meter.
#            Moreover, any combination of two inputs of the types
#            ``A 90.'', ``L 110.'', ``R 90'', and ``D 0.2'' is
#            accepted. However, the choice ``A 90.'' and ``D 0.2''
#            might be  ambiguous. By definition it fixes an
#            echelle grating with right blaze angle larger than the
#            left. To get the flipped grating with left blaze angle
#            larger than the right, the input should be ``A 90.''
#            and ``D -0.2''.
#         e.g. ``  trapezoid 60. 0.6 3 0.2 0.1 0.1 0.05''
#         -> TRAPEZOIDAL GRATING (trapezoid with the basis
#            parallel to the direction of the periodicity)
#            with angle of 60 degrees (angle enclosed by
#            basis and the sides) with a base of length 0.6
#            micro meter consisting of 3 material layers of
#            heights 0.2, 0.1, and 0.1 micro meter,
#            respectively, and with a coated layer of height
#            0.05 micro meter (greater or equal to zero)
#         e.g. ``  mtrapezoid
#                  3
#                  0.005 0.015 0.005
#                  80.   90.   80.
#                  0.01
#                  7
#                  0.05 0.075 0.05 0.05 0.075 0.075 0.05
#                  0.15 0.3   0.4  0.5  0.6   0.75  0.9  ''
#         -> MULTITRAPEZOIDAL GRATING (finite number of symmetric
#            trapezoidal bridges located side by side)
#            each bridge consists of 3 trapezoidal layers, one over
#            each other; hights in micro meter of these layers from
#            above to below are 0.005, 0.015, and 0.005; sidewall
#            angles in degrees of these layers from above to below
#            are 80, 90, 80; height in micro meter at which the
#            lateral width of the bridge is given is 0.01; number of
#            trapezoidal bridges per period is 7; lateral widths in
#            micro meter of these bridges are 0.05, 0.075, 0.05, 0.05,
#            0.075, 0.075, and 0.05; x-coordinates in micro meter of
#            the mid-points of the bottom lines of these bridges are
#            0.15, 0.3, 0.4, 0.5, 0.6, 0.75, and 0.9.
#         e.g. ``  lAmellar 3 4
#                  0.2 0.6
#                 -0.2 1.0
#                0.   0.5   0.7
#                0.0  0.50  0.90
#                0.00 0.500 0.900  ''
#       -> LAMELLAR GRATING (rectangular grating consisting
#          of several materials placed in rectangular sub-
#          domains) with 3 columns each divided into 4
#          rectangular layers, first column with x coordinate
#          in 0<x<0.2 (given in micro meter), second column with 0.2<x<0.6,
#          third columnn with 0.6<x<period (period given above),
#          whole grating with y coordinate s.t. -0.2<y<1.0,
#          first column: first layer with -0.2<y<0., second with
#          0.<y<0.5, third with 0.5<y<0.7 and fourth with 0.7<y<1,
#          second column: first layer with -0.2<y<0.0, second with
#          0.0<y<0.50, third with 0.50<y<0.90 and fourth with 0.90<y<1.,
#          third column: first layer with -0.2<y<0.00, second with
#          0.00<y<0.500, third with 0.500<y<0.900 and fourth with
#          0.900<y<1.
#       e.g. ``  lAmellar 1 1
#                0.2 0.8     ''
#       -> SIMPLE LAYER (special case of lamellar grating) with
#          layer material s.t. y-coordinate satisfies 0.2<y<0.8
#          (given in micro meter).
#       e.g. ``  polygon file1''
#       -> GRATING DETERMINED BY A POLYGONAL LINE defined
#          by the data in the file with name ``../GEOMETRIES/file1''
#          (in ``../GEOMETRIES/file1'': in each line beginning without `#'
#          there should be the x- and y-coordinate of one
#          of the consecutive corner points, first point
#          with x-coordinate 0, last point with x-coordinate
#          1, same y-coordinate for first and last point,
#          all x-coordinates between 0 and 1, at least two
#          different y-coordinates, last line should be
#          ``End'')
#       e.g. ``  polygon2 file1 file2''
#       -> COATED GRATING DETERMINED BY POLYGONAL LINES,
#          i.e. grating profile line is defined by the data
#          in the file with name ``../GEOMETRIES/file1''
#          (in ``../GEOMETRIES/file1'':
#          in each line beginning without `#' there should
#          be the x- and y-coordinate of one of the
#          consecutive corner points, first point with
#          x-coordinate 0, last point with x-coordinate
#          1, same y-coordinate for first and last point,
#          all x-coordinates between 0 and 1, at least two
#          different y-coordinates, last line should be
#          ``End'') and the coated layer is enclosed between
#          the polygonal line of ``../GEOMETRIES/file1''
#          and the polygonal line of the file with name
#          ``../GEOMETRIES/file2''
#          (in ``../GEOMETRIES/file2'':
#          in each line beginning without `#' there should be
#          the x- and y-coordinate of one of the consecutive
#          corner points, first and last point must be corner
#          of first polygon, second polygon must be on left-
#           hand side of first, one to one correspondence of
#           the corners on the two polygons between first and
#           last point of second polygon, quadrilateral between
#           corresponding segments on the left of first
#           polygon, these quadrilaterals must be disjoint, last
#           line should be ‘‘End’’)
#        e.g. ‘‘  profile’’
#        -> GRATING DETERMINED BY A SMOOTH PARAMETRIC CURVE,
#           i.e. grating determined by profile line given as
#           {(fctx(t),fcty(t)):0<=t<=1}, where the functions
#           ‘‘t|->fctx(t)’’ and ‘‘t|->fcty(t)’’ are defined
#           by the ‘‘c’’-code of the file ‘‘../GEOMETRIES/profile.c’’
#        e.g. ‘‘  profile_par 2 3
#                 1
#                 0
#                 1.5
#                 0.2
#                 0.3              ’’
#        -> GRATING DETERMINED BY A SMOOTH PARAMETRIC CURVE,
#           WITH PARAMETERS,
#           i.e. grating determined by profile line given as
#           {(fctx(t),fcty(t)):0<=t<=1}, where the functions
#           ‘‘t|->fctx(t)’’ and ‘‘t|->fcty(t)’’ are defined
#           by the ‘‘c’’-code of ‘‘../GEOMETRIES/profile_par.c’’;
#           the last code uses 2 integer parameters and 3
#           real parameters named IPARaM1, IPARaM2, RPARaM1,
#           RPARaM2, RPARaM3;
#           the integer parameters take the values 1 and 0
#           following the first line of the calling sequence
#           and the real parameters take the values 1.5, 0.2,
#           and 0.3 following the integer parameter values
#           (Any number of parameters is possible for a
#           corresponding file ‘‘../GEOMETRIES/profile_par.c’’.)
#        e.g. ‘‘  profile 0.125*sin(2.*M_PI*t)’’
#        -> GRATING DETERMINED BY A SIMPLE SMOOTH FUNCTION,
#           i.e. grating determined by sine profile line given
#           as {(t,fcty(t)):0<=t<=1}, where the function
#           ‘‘t|->fcty(t)’’ is defined by the ‘‘c’’-code
#           fcty(t)=0.125*sin(2.*M_PI*t).
#           (do not use any ‘‘blank’’/‘‘space’’ in the c-code)
#        e.g. ‘‘  profile 0.5+0.5*cos(M_PI*(1.-t)) 0.25*sin(M_PI*t)’’
#        -> GRATING DETERMINED BY A SIMPLE SMOOTH PARAMETRIC CURVE,
#           i.e., grating determined by ellipsoidal profile line
#           given as {(fctx(t),fcty(t)):0<=t<=1}, where the
#           functions ‘‘t|->fcty(t)’’ and ‘‘t|->fcty(t)’’ are
#           defined by the ‘‘c’’-codes
#           fctx(t)=0.5+0.5*cos(M_PI*(1.-t)) and
#           fcty(t)=0.25*sin(M_PI*t), respectively
#           (do not use any ‘‘blank’’/‘‘space’’ in the c-codes)
#        e.g. ‘‘  profiles’’
#        -> GRATING DETERMINED BY SMOOTH PARAMETRIC CURVES,
#           i.e. grating determined by profile lines given as
#           {(fctx(j,t),fcty(j,t)):0<=t<=1}, j=1,...,n=nmb_curves,
#           where the functions ‘‘t|->fctx(j,t)’’ and
#          ``t|->fcty(j,t)'' are defined by the ``c''-code
#          of the file ``../GEOMETRIES/profiles.c''
#       e.g. ``  profiles_par 1 2
#                3
#                0.5
#                0.50             ''
#       -> GRATING DETERMINED BY SMOOTH PARAMETRIC CURVES,
#          WITH PARAMETERS,
#          i.e. grating determined by profile lines given as
#          {(fctx(j,t),fcty(j,t)):0<=t<=1}, j=1,...,n=nmb_curves,
#          where the functions ``t|->fctx(j,t)'' and
#          ``t|->fcty(j,t)'' are defined by the ``c''-code
#          of the file ``../GEOMETRIES/profiles_par.c'';
#          the last code uses 1 integer parameter and 2
#          real parameters named IPARaM1, RPARaM1, RPARaM2;
#          the integer parameter takes the value 3
#          following the first line of the calling sequence
#          and the real parameters take the values 0.5 and 0.50
#          following the integer parameter values
#          (Any number of parameters is possible for a
#          corresponding file ``../GEOMETRIES/profiles_par.c''.)
#       e.g. ``  pin''
#       -> PIN GRATING DETERMINED BY PARAMETRIC CURVE,
#          i.e. over a flat grating with surface {(x,0):0<=x<=1}
#          a material part is attached which is located between
#          {(x,0):0<=x<=1} and {(fctx(t),fcty(t)):0<=t<=1}.
#          Here {(fctx(t),fcty(t)):0<=t<=1} is a simple open
#          arc connecting (fctx(0),fcty(0))=(xmin,0) with
#          (fctx(1),fcty(1))=(1-xmin,0) such that 0<xmin<0.5
#          is a fixed number, such that 0<fctx(t)<1, 0<t<1, and
#          such that 0<fcty(t), 0<t<1. The functions fctx, fcty
#          and the parameter xmin are defined by the code in
#          ``../GEOMETRIES/pin.c''.
#       e.g. ``  cpin''
#       -> COATED PIN GRATING DETERMINED BY TWO PARAMETRIC CURVES,
#          i.e. over a flat grating with surface {(x,0):0<=x<=1}
#          a material part is attached which is located between
#          {(x,0):0<=x<=1} and {(fctx(1,t),fcty(1,t)):0<=t<=1}.
#          Here {(fctx(1,t),fcty(1,t)):0<=t<=1} is a simple open
#          arc connecting (fctx(1,0),fcty(1,0))=(xmin,0) with
#          (fctx(1,1),fcty(1,1))=(1-xmin,0) such that 0<xmin<0.5
#          is a fixed number, such that 0<fctx(1,t)<1, 0<t<1, and
#          such that 0<fcty(1,t), 0<t<1. Additionaly, a coating
#          layer is attached located between the first curve
#          {(fctx(1,t),fcty(1,t)):0<=t<=1} and a second curve
#          {(fctx(2,t),fcty(2,t)):0<=t<=1}. The last connects the
#          point (fctx(1,arg1),fcty(1,arg1))=(fctx(2,0),fcty(2,0))
#          with (fctx(1,arg2),fcty(1,arg2))=(fctx(2,1),fcty(2,1)).
#          Moreover, {(fctx(2,t),fcty(2,t)):0<=t<=1} is a simple open
#          arc above {(fctx(1,t),fcty(1,t)):0<=t<=1} such that
#          0<fctx(2,t)<1, 0<t<1. The functions fctx(1,.), fctx(2,.),
#          fcty(1,.), and fcty(2,.) and the parameters arg1, arg2,
#          and xmin are defined by the code of the file
#          ``../GEOMETRIES/cpin.c''.
#       e.g. ``  stack 3
#                profile t /##/ 0.2*sin(2.*M_PI*t)
#                2.
#                profile 0.2*sin(2.*M_PI*t)
#                1.
#                profile t /##/ 0.
#                0.                  ''
#       -> STACK GRATING,
#          i.e. a stack of 3 profile curves shifted by 2, 1, 0
#          micro meter in vertical direction. For more details, see
#          the description in the USERGUIDE.
#       e.g. ``  bOX 0.1
#                -1. 1.
#                2 3
#                { 2.*t }
#                { -0.6 }
#                { 1.+0.4*cos(2.*M_PI*t)}
#                { 0.4*sin(2.*M_PI*t) }
#                1. 0.8
#                1. 0.
#                1. -0.8 ''
#       -> BOX GRATING,
#          i.e. a box geometry of size [0.,period=2.]x[-1.,1.]
#          given in micro m (period is supposed to be set to
#          2. in previous input lines); number 0.1 behind word
#          BOX is a mesh size factor; [0.,2.]x[-1.,1.] is
#          divided by 2 curves into 3 different material parts;
#          curves are given by the c-code in following 2 times
#          two lines; first code line is x-component of first
#          curve; second code is y-component of first curve;
#          third code is x-component of second curve; etc.;
#          last three input lines define material points;
#          material with index one is located in part of box
#          separated by above curves and containing (1.,0.8);
#          material with index two is in part of box separated
#          by above curves and containing (1.,0.); etc.; Note
#          that the curves must be simple not self intersecting,
#          contained in the box; allowed intersection points
#          of two different curves are end points only; material
#          areas separated by curves must be such that area with
#          first index contains strip beneath upper boundary side
#          of box and that area with last index contains strip
#          above lower boundary side.
#       e.g. ``  rough_mls name ''
#       -> MULTILAYER SYSTEM WITH ROUGH INTERFACES,
#          i.e. nla layers + nmld times nld layers + nlb layers,
#          This system is described by the file ``name'', which
#          is contained in the directory ``../GEOMETRIES''.
#          (alternatively, the name must contain the path of the file)
#          This input file ``name'' contains the following ordered
#          data each in a separate line (comment lines begin with '#'):
#            - dummy file name
#            - width of additional layer above and below the structure
#              (must be positive or could be 'no' for no additional layer)
#             - period of grating 'per'
#             - number 'nla' of layers above multilayer system
#             - number 'nlb' of layers below multilayer system
#             - number 'nmld' of multilayers inbetween
#             - number 'nld' of layers in each multilayer
#             - number 'ncorn' of corner points in each polygonal approximation
#               (interfaces=randomly generated polygon)
#             - number 'nrand' of standard Gaussian distributed random numbers
#               needed to generate interfaces
#               (automatically created if nrand=0, otherwise 'nrand' has
#                to equal ncorn*(nla+nld*nmld+nlb))
#             - (optional) standard Gaussian distributed random numbers
#             - 'nla' widths of layers above multilayer system,
#             - 'nld' widths of layers in each multilayer,
#             - 'nlb' widths of layers below multilayer system,
#             - standard deviation 'sigma[i]' of each layer interface,
#               for i=0,...,('nla'+'nld'*'nmld'+'nlb'-1)
#             - correlation length 'corl[i]' of each layer interface,
#               for i=0,...,('nla'+'nld'*'nmld'+'nlb'-1)
#           Note that the first of the subsequent refractive indices of
#           the grating must be that of the superstrate resp. that of
#           the adjacent upper layer. If the width of the additional
#           layers is positive, then the last of the subsequent refractive
#           indices of the grating must be that of the substrate resp. that
#           of the adjacent lower layer.
#        e.g. ``  rough_mls k name ''
#        -> MULTILAYER SYSTEM WITH ROUGH INTERFACES k TIMES,
#           just like above. However, k random realizations are computed
#           and the output values are replaced by the mean values over
#           the k computations. Standard deviations are added, too.
# Grating data:
  lamellar
#######################################################################
# Number of different grating materials (N_mat).
#    This includes the material of substrate and cover material.
#    For example,
#       if Grating data is  ``  name1''
#       -> Number of materials given in file ``name1.inp''
#       if Grating data is ``  echellea ...''
#       -> N_mat = 3 with coating height >0
#          N_mat = 2 with coating height =0
#       if Grating data is  ``  echelleb ...''
#       -> N_mat = 3 with coating height >0
#          N_mat = 2 with coating height =0
#       if Grating data is  ``  echelle ...''
#       -> N_mat = 3 with coating heights >0
#          N_mat = 2 with coating heights =0
#       if Grating data is  ``  trapezoid ...''
#       -> N_mat = number of material layers +3
#                  for coating height >0
#          N_mat = number of material layers +2
#                  for coating height =0
#       if Grating data is  ``  mtrapezoid ...''
#       -> N_mat = number of material layers in each bridge +2
#       if Grating data is  ‘‘  lAmellar k m
#                            ...  ’’
#       -> N_mat = k times m plus 2
#       if Grating data is  ‘‘  polygon file1’’
#       -> N_mat = 2
#       if Grating data is  ‘‘  polygon2 file1 file2’’
#       -> N_mat = 3
#       if Grating data is  ‘‘  profile’’
#       -> N_mat = 2
#       if Grating data is  ‘‘  profile 2 3
#                            ...        ’’
#       -> N_mat = 2
#       if Grating data is  ‘‘  profile 0.125*sin(2.*M_PI*t)’’
#       -> N_mat = 2
#       if Grating data is
#       ‘‘  profile 0.5+0.5*cos(M_PI*(1.-t)) 0.25*sin(M_PI*t)’’
#       -> N_mat = 2
#       if Grating data is ‘‘  profiles’’
#       -> N_mat = n+1 with n=nmb_curves from
#                  the file ‘‘../GEOMETRIES/profiles.c’’
#       if Grating data is ‘‘  profiles ... ’’
#       -> N_mat = n+1 with n=nmb_curves from
#                  the file ‘‘../GEOMETRIES/profiles_par.c’’
#       if Grating data is ‘‘  pin ’’
#       -> N_mat = 3
#       if Grating data is ‘‘  cpin ’’
#       -> N_mat = 4
#       if Grating data is ‘‘  stack k’’
#       -> N_mat = k+1
#       if Grating data is ‘‘  bOX ...’’
#       -> N_mat = number indicated as second
#                  integer in second lind after
#                  line with code word bOX
#       if Grating data is ‘‘  rough_mls ...’’
#       -> N_mat = nla+nld+nlb+2 if width of additional
#                  layer above and below is positive
#          N_mat = nla+nld+nlb+1 if instead of the value for
#                  the width of additional layer above and below
#                  a ’no’ is given in the input file
# Number of materials:
  2
######################################################################
# Optical indices of grating materials.
#    This is c times square root of mu times epsilon.
#    If meaningful, then the refractive indices should be ordered
#    according to the location from above to below.
#    If an input file ‘‘name1.inp’’ is used, then the optical index
#    of a subdomain with the material index j is just the j-th
#    optical index following below.
#    If grating is ‘‘lAmellar ...’’, then first material is cover
#    material, last material is substrate, and all other materials
#    are ordered from left to right and inside the columns from
#    below to above.
#    For technical reasons, the index of the material adjacent to
#    the upper line of the grating structure must coincide with
#    that of the material in the adjacent upper coated layer resp.
#    in the adjacent superstrate. Similarly the index of the
#    materials adjacent to the lower line of the grating structure
#    must coincide with that of the material in the adjacent lower
#    coated layer resp. in the adjacent substrate.
#    N_mat numbers are needed.
# Optical indices:
  1.3 +i 0.
  1.6 +i 0.
######################################################################
# Number of levels (Lev).
#    In each refinement step the step size of the mesh is halved.
#    Lev refinement steps are performed.
#    (Alternatively, one can prescribe an bound for the maximal
#     error of the efficiencies. E.g. the input ‘‘e 1.’’ means
#     that the level for the computation is the smallest positive
#     integer such that all efficiencies are computed with an
#     estimated error less than 1 per cent.)
# Number:
  1
######################################################################
# End
######################################################################
```

12.5 Output file “example.res” of FEM-FULLINFO in CLASSICAL

```
           **************************************
           **************************************
           **                                  **
           **          D P O G T R             **
           **                                  **
           **************************************
           **************************************


              date ='10. Feb 2003, 09:55:17'



 Program solves Helmholtz equation for
 optical grating and TE/TM polarization:

 boundary conditions - periodic with respect to x,
                       non-local condition on {(x,y),y=y_max}
                       and {(x,y),y=y_min}
```

```
method              - FEM
partitioning        - based on Shewchuk's ``Triangle''
matrix assembly     - pdelib
solver              - pardiso


code generated with: -DCONV


=================================================================
 INPUT FILE FOR GRATING:
=================================================================

 Name of input file without extension ``.inp'':
   example
 Comments:
   This is a fantasy grid
   for the test of gen_polyx!
 Number of materials:
   4
 Minimal angle of subdivision triangles:
   20.000000
 Upper bound for mesh size:
   0.500000
 Width of additional strip above and below:
   0.200000
 Grid points:
    1:   0.000000  0.800000
    2:   0.500000  0.800000
    3:   0.000000  0.400000
    4:   0.250000  0.400000
    5:   1.000000  0.400000
    6:   0.750000  0.200000
    7:   1.000000  0.200000
    8:   0.000000  0.000000
    9:   0.250000 -0.200000
   10:   1.000000 -0.200000
   11:   0.000000 -0.600000
   12:   1.000000 -0.800000
 Triangles:
   ( 1,  3,  4), mat= 2, fac= 1.000000
   ( 4,  6,  2), mat= 2, fac= 1.000000
   ( 6,  7,  5), mat= 2, fac= 1.000000
   ( 3,  8,  4), mat= 2, fac= 1.000000
   ( 8,  9,  4), mat= 2, fac= 0.300000
   ( 4,  9,  6), mat= 2, fac= 1.000000
   ( 6, 10,  7), mat= 3, fac= 1.000000
   ( 8, 11,  9), mat= 3, fac= 1.000000
   ( 9, 12,  6), mat= 3, fac= 1.000000
   ( 6, 12, 10), mat= 3, fac= 1.000000


=================================================================
 INPUT DATA:
```

```
=================================================================

COATED LAYERS

nmb of upper layers:    2
corresponding widths:
                        (last width = last width of *.dat
                        file + width in grating geometry)
                        0.50 micro m
                        0.40 micro m

nmb of lower layers:    3
corresponding widths:
                        (first width = first width of *.dat
                        file + width in grating geometry)
                        0.40 micro m
                        0.30 micro m
                        0.20 micro m

REFRACT.INDICES

cover material:         1.00 + i 0.00
layers above grating:
                        1.10 + i 0.00
                        1.20 + i 0.00
layers below grating:
                        2.30 + i 0.00
                        2.20 + i 0.00
                        2.10 + i 0.00
substrate material:     2.00 + i 0.00

FURTHER DATA

temperature:            20.00 degrees Celsius
wave length:            0.635 micro m
angle of inc.theta:     65.00 degrees
polarization type:      TM

GRATING

grating period:         1.00 micro m
grating height:         1.60 micro m
fem grid height:        2.00 micro m
nmb of materials:       4
corr.refract.indices:
                        1.20 + i 0.00
                        1.50 + i 0.00
                        1.70 + i 0.00
                        2.30 + i 0.00
nmb of levels f.comp.:  5


=================================================================
 INFO OF SOLUTION (LEVEL=1):
```

```
=================================================================

degrees of freedom   =  813
stepsize of discr.   =  0.81789
numb.of nonzero entr.=  5859
rate of nonzero entr.=  0.886426 per cent
memory for pardiso   =  536 kB

       Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   (  0.039594,  0.181811)   e_    0 =  3.462309
 n=   -1  theta =  -15.74   ( -0.008538,  0.021079)   e_   -1 =  0.117800
 n=   -2  theta =   21.33   (  0.031693,  0.008219)   e_   -2 =  0.236279
 n=   -3  theta =   87.07   (  0.102928, -0.466177)   e_   -3 =  2.757185

 Reflected energy:            6.573574

       Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.132885,  0.438747)   e_    0 = 22.164372
 n=    1  theta =   50.41   ( -0.034493, -0.866995)   e_    1 = 56.761614
 n=   -1  theta =    7.80   ( -0.228543, -0.163526)   e_   -1 =  9.256892
 n=   -2  theta =  -10.48   ( -0.171167,  0.062429)   e_   -2 =  3.861878
 n=   -3  theta =  -29.96   ( -0.096782, -0.007596)   e_   -3 =  0.966041
 n=   -4  theta =  -54.77   ( -0.001540,  0.078023)   e_   -4 =  0.415630

 Transmitted energy:         93.426426



=================================================================
 INFO OF SOLUTION (LEVEL=2):
=================================================================

degrees of freedom   =  3197
stepsize of discr.   =  0.34705
numb.of nonzero entr.=  23379
rate of nonzero entr.=  0.228739 per cent
memory for pardiso   =  2606 kB

       Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   ( -0.004059,  0.131487)   e_    0 =  1.730540
 n=   -1  theta =  -15.74   (  0.003041, -0.001482)   e_   -1 =  0.002606
 n=   -2  theta =   21.33   ( -0.009015,  0.008718)   e_   -2 =  0.034668
 n=   -3  theta =   87.07   (  0.007285,  0.148711)   e_   -3 =  0.268177

 Reflected energy:            2.035991

       Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.439727,  0.251543)   e_    0 = 27.066021
 n=    1  theta =   50.41   (  0.521995, -0.681800)   e_    1 = 55.589973
 n=   -1  theta =    7.80   (  0.061566, -0.302391)   e_   -1 = 11.162601
 n=   -2  theta =  -10.48   ( -0.021529, -0.115618)   e_   -2 =  1.609071
 n=   -3  theta =  -29.96   ( -0.071286, -0.050452)   e_   -3 =  0.781807
 n=   -4  theta =  -54.77   ( -0.002095,  0.160324)   e_   -4 =  1.754536

 Transmitted energy:         97.964009



=================================================================
 INFO OF SOLUTION (LEVEL=3):
=================================================================

degrees of freedom   =  12628
stepsize of discr.   =  0.19522
numb.of nonzero entr.=  92896
rate of nonzero entr.=  0.058254 per cent
memory for pardiso   =  12659 kB

       Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   (  0.012070,  0.139848)   e_    0 =  1.970304
 n=   -1  theta =  -15.74   (  0.003512,  0.014030)   e_   -1 =  0.047639
 n=   -2  theta =   21.33   ( -0.002098,  0.002527)   e_   -2 =  0.002377
 n=   -3  theta =   87.07   ( -0.079551,  0.082779)   e_   -3 =  0.159453

 Reflected energy:            2.179773

       Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.498922,  0.127814)   e_    0 = 27.975753
 n=    1  theta =   50.41   (  0.613317, -0.568916)   e_    1 = 52.762242
 n=   -1  theta =    7.80   (  0.134396, -0.299329)   e_   -1 = 12.619565
 n=   -2  theta =  -10.48   (  0.040241, -0.125325)   e_   -2 =  2.015636
 n=   -3  theta =  -29.96   (  0.038098, -0.111135)   e_   -3 =  1.414798
 n=   -4  theta =  -54.77   ( -0.101705, -0.069144)   e_   -4 =  1.032234

 Transmitted energy:         97.820227



=================================================================
 INFO OF SOLUTION (LEVEL=4):
=================================================================

degrees of freedom   =  50133
stepsize of discr.   =  0.09156
numb.of nonzero entr.=  369931
rate of nonzero entr.=  0.014719 per cent
memory for pardiso   =  59346 kB

       Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   (  0.012847,  0.143535)   e_    0 =  2.076732
 n=   -1  theta =  -15.74   (  0.000916,  0.015911)   e_   -1 =  0.057844
 n=   -3  theta =   87.07   ( -0.088879,  0.060381)   e_   -3 =  0.139668

 Reflected energy:           2.274312

       Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.502017,  0.095119)   e_    0 = 27.533702
 n=    1  theta =   50.41   (  0.636562, -0.542658)   e_    1 = 52.752053
 n=   -1  theta =    7.80   (  0.165801, -0.295567)   e_   -1 = 13.462258
 n=   -2  theta =  -10.48   (  0.064145, -0.115134)   e_   -2 =  2.020839
 n=   -3  theta =  -29.96   (  0.059586, -0.088603)   e_   -3 =  1.168650
 n=   -4  theta =  -54.77   ( -0.049270, -0.095506)   e_   -4 =  0.788186

 Transmitted energy:         97.725688



=================================================================
 INFO OF SOLUTION (LEVEL=5):
=================================================================

degrees of freedom   =  200431
stepsize of discr.   =  0.04493
numb.of nonzero entr.=  1481809
rate of nonzero entr.=  0.003689 per cent
memory for pardiso   =  278626 kB

       Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   (  0.012980,  0.144468)   e_    0 =  2.103941
 n=   -1  theta =  -15.74   (  0.000471,  0.016141)   e_   -1 =  0.059386
 n=   -2  theta =   21.33   ( -0.000145, -0.000968)   e_   -2 =  0.000211
 n=   -3  theta =   87.07   ( -0.090729,  0.055189)   e_   -3 =  0.136430

 Reflected energy:           2.299969

       Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.502603,  0.087406)   e_    0 = 27.447303
 n=    1  theta =   50.41   (  0.642427, -0.535243)   e_    1 = 52.715032
 n=   -1  theta =    7.80   (  0.174148, -0.293832)   e_   -1 = 13.675032
 n=   -2  theta =  -10.48   (  0.069855, -0.111978)   e_   -2 =  2.026452
 n=   -3  theta =  -29.96   (  0.062905, -0.082480)   e_   -3 =  1.102941
 n=   -4  theta =  -54.77   ( -0.036440, -0.097038)   e_   -4 =  0.733271

 Transmitted energy:         97.700031



=================================================================
 CONVERGENCE PROPERTIES:
=================================================================

1.VALUE (ORDER 0)
```

```
+-------+-------------+-------------+-------------+----------+
|  h    |    value    |   extrap.   |    error    | conv.ord. |
+-------+-------------+-------------+-------------+----------+
| 0.818 |    3.4623   |             |    1.3312   |          |
| 0.347 |    1.7305   |             |    0.4006   |          |
| 0.195 |    1.9703   |    1.9411   |    0.1608   |   2.85   |
| 0.092 |    2.0767   |    2.1617   |    0.0544   |   1.17   |
| 0.045 |    2.1039   |    2.1133   |    0.0272   |   1.97   |
+-------+-------------+-------------+-------------+----------+

2.VALUE (ORDER -1)

+-------+-------------+-------------+-------------+----------+
|  h    |    value    |   extrap.   |    error    | conv.ord. |
+-------+-------------+-------------+-------------+----------+
| 0.818 |    0.1178   |             |    0.0569   |          |
| 0.347 |    0.0026   |             |    0.0583   |          |
| 0.195 |    0.0476   |    0.0350   |    0.0133   |   1.36   |
| 0.092 |    0.0578   |    0.0608   |    0.0031   |   2.14   |
| 0.045 |    0.0594   |    0.0597   |    0.0015   |   2.73   |
+-------+-------------+-------------+-------------+----------+

NO ERROR ANALYSIS FOR ORDER 3

4.VALUE (ORDER -3)

+-------+-------------+-------------+-------------+----------+
|  h    |    value    |   extrap.   |    error    | conv.ord. |
+-------+-------------+-------------+-------------+----------+
| 0.818 |    2.7572   |             |    2.6240   |          |
| 0.347 |    0.2682   |             |    0.1350   |          |
| 0.195 |    0.1595   |    0.1545   |    0.0263   |   4.52   |
| 0.092 |    0.1397   |    0.1353   |    0.0065   |   2.46   |
| 0.045 |    0.1364   |    0.1358   |    0.0032   |   2.61   |
+-------+-------------+-------------+-------------+----------+

5.VALUE (REFLECTED ENERGY)

+-------+-------------+-------------+-------------+----------+
|  h    |    value    |   extrap.   |    error    | conv.ord. |
+-------+-------------+-------------+-------------+----------+
| 0.818 |    6.5736   |             |    4.2479   |          |
| 0.347 |    2.0360   |             |    0.2896   |          |
| 0.195 |    2.1798   |    2.1754   |    0.1459   |   4.98   |
| 0.092 |    2.2743   |    2.4558   |    0.0513   |   0.60   |
| 0.045 |    2.3000   |    2.3095   |    0.0257   |   1.88   |
+-------+-------------+-------------+-------------+----------+

6.VALUE (ORDER 0)

+-------+-------------+-------------+-------------+----------+
|  h    |    value    |   extrap.   |    error    | conv.ord. |
+-------+-------------+-------------+-------------+----------+
| 0.818 |     22.1644   |              |     5.1965   |           |
| 0.347 |     27.0660   |              |     0.2949   |           |
| 0.195 |     27.9758   |    28.1831   |     0.6148   |    2.43   |
| 0.092 |     27.5337   |    27.6783   |     0.1728   |    1.04   |
| 0.045 |     27.4473   |    27.4263   |     0.0864   |    2.36   |
+-------+--------------+--------------+--------------+-----------+
```

```
7.VALUE (ORDER 1)

+-------+--------------+--------------+--------------+-----------+
|  h    |     value    |    extrap.   |     error    | conv.ord. |
+-------+--------------+--------------+--------------+-----------+
| 0.818 |     56.7616   |              |     4.0836   |           |
| 0.347 |     55.5900   |              |     2.9120   |           |
| 0.195 |     52.7622   |    57.5905   |     0.0842   |   -1.27   |
| 0.092 |     52.7521   |    52.7520   |     0.0740   |    8.12   |
| 0.045 |     52.7150   |    52.7661   |     0.0370   |   -1.86   |
+-------+--------------+--------------+--------------+-----------+
```

```
8.VALUE (ORDER -1)

+-------+--------------+--------------+--------------+-----------+
|  h    |     value    |    extrap.   |     error    | conv.ord. |
+-------+--------------+--------------+--------------+-----------+
| 0.818 |      9.2569   |              |     4.6309   |           |
| 0.347 |     11.1626   |              |     2.7252   |           |
| 0.195 |     12.6196   |    17.3500   |     1.2682   |    0.39   |
| 0.092 |     13.4623   |    14.6183   |     0.4255   |    0.79   |
| 0.045 |     13.6750   |    13.7469   |     0.2128   |    1.99   |
+-------+--------------+--------------+--------------+-----------+
```

```
9.VALUE (ORDER -2)

+-------+--------------+--------------+--------------+-----------+
|  h    |     value    |    extrap.   |     error    | conv.ord. |
+-------+--------------+--------------+--------------+-----------+
| 0.818 |      3.8619   |              |     1.8298   |           |
| 0.347 |      1.6091   |              |     0.4230   |           |
| 0.195 |      2.0156   |     1.9535   |     0.0164   |    2.47   |
| 0.092 |      2.0208   |     2.0209   |     0.0112   |    6.29   |
| 0.045 |      2.0265   |     1.9495   |     0.0056   |   -0.11   |
+-------+--------------+--------------+--------------+-----------+
```

```
10.VALUE (ORDER -3)

+-------+--------------+--------------+--------------+-----------+
|  h    |     value    |    extrap.   |     error    | conv.ord. |
+-------+--------------+--------------+--------------+-----------+
| 0.818 |      0.9660   |              |     0.0712   |           |
| 0.347 |      0.7818   |              |     0.2554   |           |
| 0.195 |      1.4148   |     0.9245   |     0.3776   |   -1.78   |
| 0.092 |      1.1687   |     1.2376   |     0.1314   |    1.36   |
| 0.045 |      1.1029   |     1.0790   |     0.0657   |    1.91   |
+-------+--------------+--------------+--------------+-----------+
11.VALUE (ORDER -4)

+-------+--------------+--------------+--------------+-----------+
|  h    |    value     |    extrap.   |     error    | conv.ord. |
+-------+--------------+--------------+--------------+-----------+
| 0.818 |     0.4156   |              |     0.2627   |           |
| 0.347 |     1.7545   |              |     1.0762   |           |
| 0.195 |     1.0322   |     1.2853   |     0.3539   |    0.89   |
| 0.092 |     0.7882   |     0.6637   |     0.1098   |    1.57   |
| 0.045 |     0.7333   |     0.7173   |     0.0549   |    2.15   |
+-------+--------------+--------------+--------------+-----------+

12.VALUE (TRANSMITTED ENERGY)

+-------+--------------+--------------+--------------+-----------+
|  h    |    value     |    extrap.   |     error    | conv.ord. |
+-------+--------------+--------------+--------------+-----------+
| 0.818 |    93.4264   |              |     4.2479   |           |
| 0.347 |    97.9640   |              |     0.2896   |           |
| 0.195 |    97.8202   |    97.8246   |     0.1459   |    4.98   |
| 0.092 |    97.7257   |    97.5442   |     0.0513   |    0.60   |
| 0.045 |    97.7000   |    97.6905   |     0.0257   |    1.88   |
+-------+--------------+--------------+--------------+-----------+


=================================================================
 END:
=================================================================

      date ='10. Feb 2003, 09:56:18'

Thank you for choosing ``dpogtr''!
Bye, bye!
```

12.6 Output file “example.res” of GFEM in CLASSICAL

```
             date ='10. Feb 2003, 10:13:19'

          ***************************************
          ***************************************
          **                                   **
          **         G D P O G T R             **
          **                                   **
          ***************************************
          ***************************************


=================================================================
 INFO OF SOLUTION (LEVEL=1):
=================================================================


      Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   (  0.013025,  0.144735)   e_    0 =  2.111777
 n=   -1  theta =  -15.74   (  0.000363,  0.016237)   e_   -1 =  0.060075
 n=   -2  theta =   21.33   (  0.000035, -0.001347)   e_   -2 =  0.000400
 n=   -3  theta =   87.07   ( -0.091483,  0.053904)   e_   -3 =  0.136395

 Reflected energy:            2.308647

      Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.502786,  0.085087)   e_    0 = 27.424602
 n=    1  theta =   50.41   (  0.644183, -0.532792)   e_    1 = 52.687989
 n=   -1  theta =    7.80   (  0.176698, -0.293270)   e_   -1 = 13.741238
 n=   -2  theta =  -10.48   (  0.071550, -0.111089)   e_   -2 =  2.031272
 n=   -3  theta =  -29.96   (  0.063904, -0.080675)   e_   -3 =  1.085740
 n=   -4  theta =  -54.77   ( -0.032720, -0.097400)   e_   -4 =  0.720512

 Transmitted energy:         97.691353


=================================================================
 INFO OF SOLUTION (LEVEL=2):
=================================================================


      Reflection efficiencies and coefficients

 n=    0  theta =  -65.00   (  0.013043,  0.144752)   e_    0 =  2.112318
 n=   -1  theta =  -15.74   (  0.000365,  0.016240)   e_   -1 =  0.060098
 n=   -2  theta =   21.33   (  0.000044, -0.001384)   e_   -2 =  0.000423
 n=   -3  theta =   87.07   ( -0.091547,  0.053854)   e_   -3 =  0.136472

 Reflected energy:            2.309310

      Transmission efficiencies and coefficients

 n=    0  theta =   26.95   ( -0.502838,  0.084941)   e_    0 = 27.427416
 n=    1  theta =   50.41   (  0.644282, -0.532607)   e_    1 = 52.682703
 n=   -1  theta =    7.80   (  0.176861, -0.293228)   e_   -1 = 13.745111
 n=   -2  theta =  -10.48   (  0.071650, -0.111036)   e_   -2 =  2.031559
 n=   -3  theta =  -29.96   (  0.063954, -0.080553)   e_   -3 =  1.084372
 n=   -4  theta =  -54.77   ( -0.032481, -0.097406)   e_   -4 =  0.719528
 Transmitted energy:         97.690690


=================================================================
 END:
=================================================================

      date ='10. Feb 2003, 10:30:53'

Thank you for choosing ``gdpogtr''!
Bye, bye!
```

12.7 Output file "example.res" of FEM in CONICAL

```
             date ='10. Feb 2003, 10:06:22'

          ***************************************
          ***************************************
          **                                   **
          **          C O N I C A L            **
          **                                   **
          ***************************************
          ***************************************


=====================================================================
 INFO OF SOLUTION (LEVEL=1):
=====================================================================



       Reflection efficiencies and coefficients
       ----------------------------------------

 Order   Phi    Theta           E_z
                                H_z           Efficiency
     0   47.00  30.00  (  0.19657, -0.16355)
                       ( -0.05050, -0.15697)    9.25813
    -1  128.80  27.98  (  0.01482, -0.03567)
                       (  0.06213, -0.13130)    2.30360
    -2  158.51  86.74  (  0.07634,  0.09663)
```

```
                    (  0.12963,  0.05033)    0.22670

 Reflected energy:     11.78843

       Transmission efficiencies and coefficients
       ------------------------------------------

 Order   Phi    Theta           E_z
                                H_z           Efficiency
    0   47.00 160.53  ( -0.18113,  0.45695)
                      ( -0.14912,  0.68850)   80.24261
    1   20.54 135.99  (  0.07497,  0.15146)
                      ( -0.05129,  0.15348)    5.32346
   -1  128.80 161.77  ( -0.05523, -0.08440)
                      (  0.02083, -0.07834)    2.29025
   -2  158.51 138.27  (  0.03830,  0.01953)
                      (  0.01062,  0.03930)    0.35525

 Transmitted energy:   88.21157


=====================================================================
 INFO OF SOLUTION (LEVEL=2):
=====================================================================



       Reflection efficiencies and coefficients
       ----------------------------------------

 Order   Phi    Theta           E_z
                                H_z           Efficiency
    0   47.00  30.00  (  0.00901, -0.00844)
                      ( -0.13596, -0.02289)    1.91613
   -1  128.80  27.98  ( -0.00200,  0.08454)
                      (  0.04245,  0.01373)    0.93218
   -2  158.51  86.74  ( -0.06824,  0.05857)
                      ( -0.06300,  0.12368)    0.17972

 Reflected energy:      3.02802

       Transmission efficiencies and coefficients
       ------------------------------------------

 Order   Phi    Theta           E_z
                                H_z           Efficiency
    0   47.00 160.53  ( -0.31129,  0.27182)
                      ( -0.49973,  0.42367)   62.77211
    1   20.54 135.99  ( -0.26115,  0.05446)
                      ( -0.32171,  0.20006)   17.87244
   -1  128.80 161.77  ( -0.12492, -0.19255)
                      ( -0.13873, -0.20804)   14.07546
   -2  158.51 138.27  (  0.08036, -0.05007)
```

```
                      (  0.10289, -0.07817)    2.25198

 Transmitted energy:   96.97198


======================================================================
 INFO OF SOLUTION (LEVEL=3):
======================================================================




       Reflection efficiencies and coefficients
       ----------------------------------------

 Order   Phi    Theta           E_z
                                H_z           Efficiency
     0   47.00  30.00  (  0.06129, -0.00846)
                       ( -0.09079, -0.04955)    1.45263
    -1  128.80  27.98  ( -0.02946,  0.01247)
                       (  0.02120, -0.02318)    0.20498
    -2  158.51  86.74  ( -0.05922, -0.00385)
                       ( -0.01123,  0.00823)    0.02441

 Reflected energy:      1.68202

       Transmission efficiencies and coefficients
       ------------------------------------------

 Order   Phi    Theta           E_z
                                H_z           Efficiency
     0   47.00 160.53  ( -0.38108,  0.19799)
                       ( -0.58998,  0.31418)   66.49400
     1   20.54 135.99  ( -0.22472, -0.00453)
                       ( -0.37512,  0.09299)   15.48188
    -1  128.80 161.77  ( -0.07243, -0.23512)
                       ( -0.06585, -0.25190)   15.85666
    -2  158.51 138.27  (  0.05169,  0.00215)
                       (  0.03000, -0.03203)    0.48544

 Transmitted energy:   98.31798


======================================================================
 INFO OF SOLUTION (LEVEL=4):
======================================================================




       Reflection efficiencies and coefficients
       ----------------------------------------

 Order   Phi    Theta           E_z
```

```
                               H_z          Efficiency
    0   47.00  30.00  (  0.06068,  0.01649)
                      ( -0.07617, -0.03097)    1.07145
   -1  128.80  27.98  ( -0.02951,  0.00963)
                      (  0.01652, -0.00861)    0.13367
   -2  158.51  86.74  ( -0.04706, -0.03015)
                      (  0.00604,  0.01967)    0.02330

 Reflected energy:      1.22842

       Transmission efficiencies and coefficients
       -----------------------------------------

 Order   Phi    Theta          E_z
                               H_z          Efficiency
    0   47.00 160.53  ( -0.38965,  0.12856)
                      ( -0.61617,  0.20649)   61.81559
    1   20.54 135.99  ( -0.25045, -0.03050)
                      ( -0.41925,  0.03884)   18.86494
   -1  128.80 161.77  ( -0.02063, -0.25490)
                      ( -0.01257, -0.27967)   17.53158
   -2  158.51 138.27  (  0.05192,  0.01214)
                      (  0.04725, -0.02301)    0.55947

 Transmitted energy:   98.77158


====================================================================
 INFO OF SOLUTION (LEVEL=5):
====================================================================




       Reflection efficiencies and coefficients
       ---------------------------------------

 Order   Phi    Theta          E_z
                               H_z          Efficiency
    0   47.00  30.00  (  0.06175,  0.01688)
                      ( -0.07173, -0.03108)    1.02091
   -1  128.80  27.98  ( -0.02812,  0.00775)
                      (  0.01579, -0.00606)    0.11595
   -2  158.51  86.74  ( -0.04494, -0.03733)
                      (  0.00386,  0.01787)    0.02462

 Reflected energy:      1.16148

       Transmission efficiencies and coefficients
       -----------------------------------------

 Order   Phi    Theta          E_z
                               H_z          Efficiency
    0   47.00 160.53  ( -0.39161,  0.10975)
```

```
                     ( -0.61958,  0.17733)   60.76400
      1   20.54 135.99  ( -0.25177, -0.03911)
                     ( -0.42754,  0.02212)   19.38467
     -1  128.80 161.77  ( -0.00565, -0.25896)
                     (  0.00395, -0.28489)   18.04542
     -2  158.51 138.27  (  0.05587,  0.01457)
                     (  0.05139, -0.02021)    0.64442

  Transmitted energy:   98.83852


 ======================================================================
  END:
 ======================================================================

       date ='10. Feb 2003, 10:06:25'

 Thank you for choosing ``CONICAL''!
 Bye, bye!
```

12.8 Data file "example.dat" of OPTIMIZE in OPTIM

```
#-*-makefile-*-
######################################################################
#                                                                    #
#                     #####################                          #
#                     #####################                          #
#                     ##                 ##                          #
#                     ##  E X A M P L E  ##                          #
#                     ##                 ##                          #
#                     #####################                          #
#                     #####################                          #
#                                                                    #
#         all lines beginning with ``#'' are comments!               #
#                                                                    #
######################################################################
#
#               input file for local optimization
#               using GFEM for conical diffraction
#
######################################################################
#                                                                    #
#              N A M E   O F   O U T P U T   F I L E                 #
#                                                                    #
######################################################################
# Name of the output file.
#    The tag ``.res'' will be added.
#    The file will be written in the ``RESULTS'' directory.
#    (Alternatively, a path for the location of the file
#     can be added before the name. This must contain at
#     least one slash '/'. E.g. for a file ``name.res''
#     in the current working directory write ``  ./name'')
# Name:
  example
######################################################################
#                                                                    #
#             G R A T I N G   +   I L L U M I N A T I O N            #
#                                                                    #
######################################################################
# Number of coating layers over the grating (N_co_ov).
#    The grating cross section consists of a rectangular area
#    parallel to the axes. This inhomogeneous part is determined
#    by a triangular grid and can have already a few
#    layers of coatings involved. Beneath and above
#    this rectangular structure, there might be additional
#    coated layers of rectangular shape. These kind of
#    layers are called coating layers over the grating and
#    coating layers beneath the grating, respectively.
#    Alternatively a
#    multilayer system input format is possible:
#    E.g. the input ``MLS n1 n2*n3 n4'' with n1,n2,n3,n4
#    replaced by non-negative integers means
#    N_co_ov=n1+n2*n3+n4 layers with n1 layers above,
#    n2 groups of n3 layers with same widths and materials
#    in the middle, and with 7 layers below.
# Number:
  0
######################################################################
# Widths of coating layers in micro meter.
#    N_co_ov entries. Needed only if N_co_ov >0.
#    Else no entry and no line.
# Widths:
######################################################################
# Number of coating layers beneath the grating (N_co_be):
  0
######################################################################
# Widths of coating layers in micro meter.
#    N_co_be entries. Needed only if N_co_be >0.
#    Else no entry and no line.
#    For a multilayer system the widths of the
#    n3 layers in the groups must be given only once.
#    I.e. for a multilayer system n1+n3+n4 input numbers
#    are needed.
# Widths:
######################################################################
# Wave length in micro meter (lambda).
#    Either add a single value e.g. ``.63''.
#    Either add more values by e.g.
#      ``  V
#          5
#          .63
#          .64
#          .65
#          .69
#          .70  ''.
#      The last means that computation is to be done for
#      the wave lengths from the Vector of length 5:
#      ``.63'', ``.64'',``.65'',``.69'', and ``.70''.
#    Or add e.g.
#      ``  I .63 .73 .02''.
#      The last means that computation is to be done for
#      the wave lengths ``.63+i*.02'' with i=0,1,2,... and with
#      wave length ``.63+i*.02'' less or equal to ``.73''.
# Wave length:
#  1.2
  I .635 .636 .002
######################################################################
# Temperature in degrees Celsius from 0 to 400.
#    20. for room temperature!
#    Must be set to any fixed number.
#    Will be ignored if optical indices are given explicitly.
# Temperature:
  20.
######################################################################
# Optical index (refractive index) of cover material.
#    This is c times square root of mu times epsilon.
#    This could be complex like e.g. ``4.298 +i 0.073'' for
#    Si with wave length 500nm.
#    This could be also given by the name of a material
#    like: Air Ag Al Au CsBr Cu InP MgF2 NaCl PMMA PSKL
#          SF5 Si TlBr TlCl Cr ZnS Ge Si1.0 - Si2.0
#          TiO2r Quarz AddOn ... (cf. Userguide)
#    This could be a value interpolated from a user
#    defined table, determined by the name of the file
#    (file is to be located in the current directory,
#    name of file must begin with letter ``u'' and may
#    consist of no more than five letters like e.g. user,
#    the file consists of lines each with three real
#    numbers, first: wave length in micro meter, second:
#    the real part of the corresponding optical index,
#    third: the imaginary part of the corresponding index).
# Optical index:
  1.0 +i 0.
######################################################################
# Optical indices of the materials of the upper coating layers.
#    This is c times square root of mu times epsilon.
#    N_co_ov entries. Needed only if N_co_ov >0.
#    Else no entry and no line.
#    For a multilayer system the indices of the
#    n3 layers in the groups must be given only once.
#    I.e. for a multilayer system n1+n3+n4 input lines
#    are needed.
# Optical indices:
######################################################################
```

```
# Optical indices of the materials of the lower coating layers.
#    This is c times square root of mu times epsilon.
#    N_co_be entries. Needed only if N_co_be >0.
#    Else no entry and no line.
# Optical indices:
######################################################################
# Optical index of substrate material.
#    This is c times square root of mu times epsilon.
# Optical index:
  1.5 +i 0.
######################################################################
# Type of output results.
#    Either ‘‘TE/TM’’: results in terms of TE and TM part of Wave.
#    Either ‘‘Jones’’: results in terms of Jones vector
#                      representation.
#    Or ‘‘3.Com’’:     results in terms of the component in the z
#                      axis, that is in the direction of the grooves.
#    For more details cf. Section 2.3 in USERGUIDE.ps.
# Type:
  TE/TM
######################################################################
# Type of polarization and coordinate system for incoming wave vector.
#    Either ‘‘TE’’: means that incident electric field is perpendicular
#                   to wave vector and to normal of grating plane
#                   (plane of grating grooves) and
#                   incoming wave vector is presented in xz system as
#                   (sin theta cos phi, -cos theta, sin theta sin phi).
#    Either ‘‘TM’’: means that incident magnetic field is perpendicular
#                   to wave vector and to normal of grating plane and
#                   incoming wave vector is presented in xz system as
#                   (sin theta cos phi, -cos theta, sin theta sin phi).
#    Either ‘‘TE/TM’’:
#                   means TE and TM, two calculations.
#    Either ‘‘TP’’: means polarized electro-magnetic field and
#                   incoming wave vector is presented in xz system as
#                   (sin theta cos phi, -cos theta, sin theta sin phi).
#    Or ‘‘pol’’:    means polarized electro-magnetic field and
#                   incoming wave vector is presented in xy system as
#                   (sin theta cos phi, -cos theta cos phi, sin phi).
# Type:
  TP
######################################################################
# Parameter of polarization.
#    If type of polarization is ‘‘pol’’ or ‘‘TP’’, then this is
#    the angle (in degrees) between x axis (axis in plane of
#    grating grooves which is perpendicular to grooves)
#    and projection of electric field vector onto x-z plane of
#    grating grooves.
#    Needed only if polarization is of type ‘‘pol’’ or ‘‘TP’’.
#    Else no entry and no line.
# Parameter:
  20.
######################################################################
# Angle of incident wave in degrees (theta).
#    If type of polarization is ‘‘pol’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta cos phi, sin phi)
#    with the restriction -90 < phi,theta < 90.
#    If type of polarization is ‘‘TE’’/‘‘TM’’/‘‘TP’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta, sin theta sin phi)
#    with the restriction 0 < theta < 90.
#    Either add a single value e.g. ‘‘45.’’.
#    Either add more values by e.g.
#      ‘‘  V
#          5
#          63.
#          64.
#          65.
#          69.
#          70. ’’.
#      The last means that computation is to be done for
#      the angles from the Vector of length 5:
#      ‘‘63.’’, ‘‘64.’’,‘‘65.’’,‘‘69.’’, and ‘‘70.’’.
#    Or add e.g.
#      ‘‘  I 45 56 2’’.
#      The last means that computation is to be done for
#      the angles ‘‘45+i*2’’ with i=0,1,2,... and with
#      angle ‘‘45+i*2’’ less or equal to ‘‘56’’.
#    Note that either the wave length or the angle of
#    incident wave theta must be single valued.
# Angle:
  I 30. 31. 2.
######################################################################
# Angle of incident wave in degrees (phi).
#    If type of polarization is ‘‘pol’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta cos phi, sin phi)
#    with the restriction -90 < phi,theta < 90.
#    If type of polarization is ‘‘TE’’/‘‘TM’’/‘‘TP’’,
#    then the incident light beam takes the direction
#    (sin theta cos phi, -cos theta, sin theta sin phi)
#    with the restriction 0 < theta < 90.
#    Either add a single value e.g. ‘‘45.’’.
#    Either add more values by e.g.
#      ‘‘  V
#          5
#          63.
#          64.
#          65.
#          69.
#          70. ’’.
#      The last means that computation is to be done for
#      the angles from the Vector of length 5:
#      ‘‘63.’’, ‘‘64.’’,‘‘65.’’,‘‘69.’’, and ‘‘70.’’.
#    Or add e.g.
#      ‘‘  I 45 56 2’’.
#      The last means that computation is to be done for
#      the angles ‘‘45+i*2’’ with i=0,1,2,... and with
#      angle ‘‘45+i*2’’ less or equal to ‘‘56’’.
#    Note that two of the three, the wave length, the
#    angle of incident wave theta, and the angle of
#    incident wave phi, must be single valued.
# Angle:
  I 47. 48. 2.
####################################################################
# Length factor of additional shift of grating geometry.
#    This is shift into the x-direction, i.e. the
#    direction of the period to the right.
#    This is length of shift relative to period, i.e.
#    the grating structure given by subsequent input
#    will be shifted by factor times the period given
#    in subsequent input.
#    However, only the Rayleigh numbers and efficiencies
#    will be computed according to the shift. The field
#    vectors in the plots are drawn without shift, and
#    the graphics of the executable with tag ‘‘_CHECK’’
#    is drawn without shift!
#    Must be a real number between 0 and 1.
# Length:
  0.
####################################################################
# Stretching factor for grating in y-direction.
#    Must be a positive real number.
# Length:
  1.
####################################################################
# Length of additional shift of grating geometry in micro m.
#    This is shift into the y-direction, i.e. the
#    direction perpendicular to the grating surface
#    pointing into the cover material.
#    Must be a real number.
# Length:
  0.
#######################################################################
# Period of grating in micro meter:
  0.25
#######################################################################
#                                                                     #
#           P A R A M E T E R S   O F   G R A T I N G                 #
#                                                                     #
#######################################################################
# Parameters nd_geom_param, dl_geom_param, du_geom_param, and
# ni_geom_param, and i_geom_param to describe grating geometry.
#
#    integer parameters:        i_geom_param[i],
#                               i=1,...,ni_geom_param
#                               In particular
#                               i_geom_param[1] - class of gratings
#                               i_geom_param[2] - number of materials
#    real parameters:           d_geom_param[i],
#                               i=1,...,nd_geom_param
#    bounds of real parameters: dl_geom_param[i],du_geom_param[i],
#                               i=1,...,nd_geom_param,
#                               dl_geom_param[i]<=du_geom_param[i].
#    name parameters (character strings):
#                               s_geom_param[i],
#                               i=1,...,ns_geom_param
#
#    Optimization runs over all real parameters d_geom_param[i] in
#    [dl_geom_param[i],du_geom_param[i]] with i such that
#    dl_geom_param[i]<du_geom_param[i].The parameters d_geom_param[i]
#    with dl_geom_param[i]=du_geom_param[i] keep their values.
#    The integer and string parameters switch between the implemented
#    sub classes of gratings.
#
#    If the real parameter is the real resp. imaginary part of a
#    refractive index, then its value can be fixed by an input string
#    like ‘‘Re Cr’’ or ‘‘Im Ag’’. The same string must be the input
#    for the corresponding lower and upper bounds (cf. Userguide).
#
#    If a real parameter depends on other free parameters, then this
#    dependency can be indicated as follows. Write ‘‘Dep’’ on the
#    input place for d_geom_param[i] and du_geom_param[i]. On the
#    input place of dl_geom_param[i] write ‘‘Dep:’’ followed by the
#    dependency in c-code. For instance, the simple dependency
#    expression d_geom_param[i]=d_geom_param[11]*d_geom_param[7] is
#    indicated by the line (cf. Userguide):
#                                            Dep: p11*p7
#
#    If possible try to present the dependency in the form
#
#                                            Dep: f1+(f2)/(f3)
#
#    with f1, f2, and f3 as terms including the variables
#    d_geom_param[i], constants, and the operations +, -, and *
#    but without any blank and bracket.
#
##################
#
# CLASS 1 (i_geom_param[1]=1): polygonal function profile
# ---------------------------------------------------------
#
#       - Polygonal grating defined by profile function
#         which is piecewise linear between the knots over
#         a uniform partition of the interval [0,Period]
#       - Number of string parameters:  ns_geom_param=0
#       - Number of integer parameters: ni_geom_param=3
#              i_geom_param[1]=1
#              i_geom_param[2]=2
#              i_geom_param[3]=number of interior knots in the
#                              uniform partition of [0,Period]
#       - Number of real parameters:    nd_geom_param=i_geom_param[3]+4
#              k=1,...,i_geom_param[3]
#              d_geom_param[k]=function value of profile curve
#                              at Period*k/(i_geom_param[3]+1),
#                              in other words, polygonal grating
#                              has i_geom_param[3]+2 corners at
#                              the points
#                                  (0,0)
#                                  (x_k,y_k), k=1,...,i_geom_param[3]
#                                  (Period,0)
#                              where:
#                                   x_k:=Period*k/(i_geom_param[3]+1)
#                                   y_k:=d_geom_param[k]
#              k=i_geom_param[3]+1
#              d_geom_param[k]=real part of refractive index of
#                              cover material
#              k=i_geom_param[3]+2
#              d_geom_param[k]=imaginary part of refractive index of
#                              cover material
#              k=i_geom_param[3]+3
#              d_geom_param[k]=real part of refractive index of
#                              substrate material
#              k=i_geom_param[3]+4
#              d_geom_param[k]=imaginary part of refractive index of
#                              substrate material
#       - Following parameters must be fixed by setting upper bound=
#         lower bound:
#              d_geom_param[k],k=i_geom_param[3]+1,...,i_geom_param[3]+4
#
##################
#
# CLASS 2 (i_geom_param[1]=2): polygonal profile curve, no proper constraints
# ---------------------------------------------------------------------------
#
#       - Polygonal grating defined by profile curve
#         which is piecewise linear between the knots
#       - Warning: iterative solutions are tested for
#                  selfintersection
#                  but the constraints expressed by no
#                  selfintersection is not included into
#                  the optimization
#       - Number of string parameters:  ns_geom_param=0
#       - Number of integer parameters: ni_geom_param=3
#              i_geom_param[1]=2
#              i_geom_param[2]=2
#              i_geom_param[3]=number of interior knots
#                              with x-coordinate in (0,Period)
#       - Number of real parameters:    nd_geom_param=2*i_geom_param[3]+4
#              m=1,...,i_geom_param[3]
#              (d_geom_param[2*m-1],d_geom_param[2*m])
#                             =corner point of profile curve
#                              in other words, polygonal grating
#                              has i_geom_param[3]+2 corners at
#                              the points
#                                  (0,0)
#                                  (x_m,y_m), m=1,...,i_geom_param[3]
#                                  (Period,0)
#                              where:
```

```
#                                  x_m:=d_geom_param(2*m-1)
#                                  y_m:=d_geom_param(2*m)
#              k=2*i_geom_param[3]+1
#              d_geom_param[k]=real part of refractive index of
#                              cover material
#              k=2*i_geom_param[3]+2
#              d_geom_param[k]=imaginary part of refractive index of
#                              cover material
#              k=2*i_geom_param[3]+3
#              d_geom_param[k]=real part of refractive index of
#                              substrate material
#              k=2*i_geom_param[3]+4
#              d_geom_param[k]=imaginary part of refractive index of
#                              substrate material
#       - Following parameters must be fixed by setting upper bound=
#         lower bound:
#              d_geom_param[k],k=2*i_geom_param[3]+1,...,2*i_geom_param[3]+4
#
##################
#
# CLASS 3 (i_geom_param[1]=3): polygonal profile curve with proper constraints
# ----------------------------------------------------------------------------
#
#       - Polygonal grating defined by profile curve
#         which is piecewise linear between the knots
#       - In contrast to CLASS 2:
#                constraints expressed by no
#                selfintersection (cf. the constraints below)
#                is included into the optimization
#       - For this class, we have implemented only the conjugate
#         gradient algorithm (choose ind_opt=1).
#       - Number of string parameters:  ns_geom_param=0
#       - Number of integer parameters: ni_geom_param=3
#              i_geom_param[1]=3
#              i_geom_param[2]=2
#              i_geom_param[3]=nkn:=number of interior knots
#                                  with x-coordinate in (0,Period)
#       - Number of real parameters:    nd_geom_param=2*i_geom_param[3]+5
#              k=1,...,i_geom_param[3]
#              P_k:=(d_geom_param[2*m-1],d_geom_param[2*m])
#                             =corner point of profile curve
#                              in other words, polygonal grating
#                              has i_geom_param[3]+2 corners at
#                              the points
#                                  P_0      :=(0,0),
#                                  P_m      :=(x_m,y_m), m=1,...,nks
#                                  P_{nks+1}:=(Period,0)
#                              where:
#                                  x_m:=d_geom_param(2*m-1)
#                                  y_m:=d_geom_param(2*m)
#                                  nks:=i_geom_param[3]
#              k=2*i_geom_param[3]+1
#              d_geom_param[k]=real part of refractive index of
#                              cover material
#              k=2*i_geom_param[3]+2
#              d_geom_param[k]=imaginary part of refractive index of
#                              cover material
#              k=2*i_geom_param[3]+3
#              d_geom_param[k]=real part of refractive index of
#                              substrate material
#              k=2*i_geom_param[3]+4
#              d_geom_param[k]=imaginary part of refractive index of
#                              substrate material
#              k=2*i_geom_param[3]+5
#              d_geom_param[k]=small threshold EPSilon appearing
#                              in the constraint conditions for the
#                              feasible sets of parameters
#                              (0.0001<EPSilon<0.5)
#       - Constraints:
#              first nks+1 conditions:      two consecutive points not too
#                                           close to each other
#              next nks*(nks-1) conditions: each point is not to close to
#                                           each side of polygonal which
#                                           does not contain the point
#                                           (point not in ellipse around
#                                           side with small half axis
#                                           about square root of EPSilon
#                                           times side length)
#              last condition:              no intersection of non-neighbour
#                                           lines
#              more precisely:
#              a) i=1,2,...,nks+1
#
#                              2         2       2
#                 |P_i-P_(i-1)| >=EPSilon *Period
#
#              b) j=1,2,...,nks,
#                 m=1,2,...,j-1,j+2,...,nks+1,
#                 i=nks+1+(m'-1)*nks+j, m':=m   if m<j
#                                       m':=m-2 if m>j
#
#                 |P_j-P_m|+|P_j-P_(m-1)|-|P_m-P_(m-1)|>=EPSilon*|P_m-P_(m-1)|
#
#              c) no intersection of [P_(i-1),P_i] and [P_(j-1),P_j] for
#                 i,j=1,2,3,...,nks+1 if |i-j|>1
#       - Following parameters must be fixed by setting upper bound=
#         lower bound:
#              d_geom_param[k],k=2*i_geom_param[3]+1,...,2*i_geom_param[3]+5
#
##################
#
# CLASS 4 (i_geom_param[1]=4): stack of trapezoids
# ------------------------------------------------
#
#       - Stack grating consisting of several trapezoids
#         with refractive indices included into the set of
#         optimization parameters
#       - Number of trapezoids is prescribed
#       - Whole stack in one period of the grating
#       - Lower side of stack is fixed by two parameters:
#           param_1 = posb/d    = ratio of distance of right lower corner
#                                 from the left boundary line of the period
#                                 and period d
#           param_2 = posa/posb = ratio of distance of left lower corner
#                                 from the left boundary line of the period
#                                 and distance of right lower corner
#                                 from the left boundary line of the period
#       - Each trapezoid is determined by its hight, by the lower side
#         which is the upper side of the adjacent lower trapezoid, and
#         by the upper side prescribed by the two parameters:
#           param_k_1 = b_k/d   = ratio of distance of right upper corner
#                                 from the left boundary line of the period
#                                 and period d
#           param_k_2 = a_k/b_k = ratio of distance of left upper corner
#                                 from the left boundary line of the period
#                                 and distance of right upper corner
#                                 from the left boundary line of the period
#       - Refractive index of the material of each trapezoid
#         is prescribed as an optimization parameter
#       - Number of string parameters:  ns_geom_param=0
#       - Number of integer parameters: ni_geom_param=2
#              i_geom_param[1]=4
#              i_geom_param[2]=number of different materials
#                             =2+number of trapezoids
#       - Number of real parameters:    nd_geom_param=5*i_geom_param[2]-4
#
#              k=1,...,i_geom_param[2]-2
#              h_k:=d_geom_param[5*k-4]    =height of k-th trapezoid in stack
#                                           (in micro meter, h_k>0)
#              b_k/d:=d_geom_param[5*k-3]  =ratio of distance of right upper
#                                           corner from the left boundary
#                                           line of the period and period d
#                                           (0<b_k/d<=1)
#              a_k/b_k:=d_geom_param[5*k-2]=ratio of distance of left upper
#                                           corner from the left boundary
#                                           line of the period and distance
#                                           of right upper corner from the
#                                           left boundary line of the period
#                                           (0<=a_k/b_k<1.
#                                            If not both of the parameters
#                                            b_k/d and a_k/b_k are fixed,
#                                            then we require:
#                                            a_k/b_k>0 for b_k/d=1)
#              n_k:=d_geom_param[5*k-1]+i*d_geom_param[5*k]
#                                          =refractive index
#                                           of k-th trapezoid in stack
#                                           (Re n_k>0, Im n_k>=0)
#              k=5*i_geom_param[2]-9
#              posb/d:=d_geom_param[k]     =ratio of distance of right lower
#                                           corner from the left boundary
#                                           line of the period and period d
#                                           (0<posb/d<=1)
#              k=5*i_geom_param[2]-8
#              posa/posb:=d_geom_param[k]  =ratio of distance of left lower
#                                           corner from the left boundary
#                                           line of the period and distance
#                                           of right lower corner from the
#                                           left boundary line of the period
#                                           (0<=posa/posb<1.
#                                            If not both of the parameters
#                                            posb/d and posa/posb are fixed,
#                                            then we require:
#                                            posa/posb>0 for posb/d=1)
#              k=5*i_geom_param[2]-7
#              n_co:=d_geom_param[k]+i*d_geom_param[k+1]
#                                          =refractive index of cover
#                                           material
#                                           (Re n_co>0, Im n_co>=0)
#              k=5*i_geom_param[2]-5
#              n_su:=d_geom_param[k]+i*d_geom_param[k+1]
#                                          =refractive index of substrate
#                                           material resp. material
#                                           of adjacent lower coating
#                                           strip
#                                           (Re n_su>0, Im n_su=0)
#       - Following parameters must be fixed by setting upper bound=
#         lower bound:
#              d_geom_param[k],k=5*i_geom_param[2]-7,...,5*i_geom_param[2]-4
#
#
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Refractive indices:
#
#                                        n_co
#
#                            +------------------------------+
#                           /            n_3                |
#                          /                                |
#                         +---------------------------------+
#                          \                               /
#                           \                             /
#                            \           n_2             /
#                             \                         /
#                              \                       /
#                               \                     /
#                                +------------------+
#                               /        n_1         \
#              +----------------+----------------------+------------------+
#
#                                       n_su
#
#
```

```
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Geometry parameters:
#
#                                    b_k
#              <............................................>
#              .                                            .
#              .   a_k                                      .
#              <..........>                                 .
#              .          .                                 .
#              .          .                                 .
#              .          .  +------------------------------+
#              .          . /                               |
#              .          ./                                |
#              .          +---------------------------------+
#              .           \                               /    ^
#              .            \                             /     :
#              .             \                           /      : h_k
#              . k-th trapez.:\                         /       :
#              . (k=2)         \                       /        :
#              .                \                     /         v
#              .                 +-------------------+
#              .                /                     \
#              +---------------+-----------------------+-----------------+
#            (0,0)        (posa,0)               (posb,0)              (d,0)
#
#
#
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              The presented parameters of CLASS 4 are the internal parameters.
#              The class can be determined by external parameters, too:
#
#
#                  External geometry parameters:
#
#                                         d_k
#                          <.................................>
#
#
#                             +------------------------------+
#                            /                               |
#                           /                                |
#                          +---------------------------------+
#                           \                               /    ^
#                            \                             /     :
#                             \                  .  .  .  /      : h_k
#               k-th trapezoid:\              .          /       :
#               (k=2)           \           . angle a_k /        :
```

```
#                                 \         .           /         v
#                                  +--------------------+
#                                 /                     \
#               +----------------+-----------------------+-----------------+
#             (0,0)         (posa,0)                (posb,0)             (d,0)
#
#
#
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#
#              Number of real parameters: nd_geom_param=5*i_geom_param[2]-4
#
#              k=1,...,i_geom_param[2]-2
#              h_k:=d_geom_param[5*k-4]    =height of k-th trapezoid in stack
#                                           (in micro meter, h_k>0)
#              d_k:=d_geom_param[5*k-3]    =length of the upper side
#                                            of k-th trapezoid in stack
#                                            (in micro meter)
#              a_k:=d_geom_param[5*k-2]    =angle at the right lower corner
#                                            of k-th trapezoid in stack
#                                            (in degrees)
#              n_k:=d_geom_param[5*k-1]+i*d_geom_param[5*k]
#                                          =refractive index
#                                           of k-th trapezoid in stack
#                                           (Re n_k>0, Im n_k>=0)
#              k=5*i_geom_param[2]-9
#              posa:=d_geom_param[k]       =distance of left lower stack
#                                           corner from left starting point
#                                           of the period
#                                           (in micro meter)
#              k=5*i_geom_param[2]-8
#              posb:=d_geom_param[k]       =distance of right lower stack
#                                           corner from left starting point
#                                           of the period
#                                           (in micro meter)
#              k=5*i_geom_param[2]-7
#              n_co:=d_geom_param[k]+i*d_geom_param[k+1]
#                                          =refractive index of cover
#                                           material
#                                           (Re n_co>0, Im n_co>=0)
#              k=5*i_geom_param[2]-5
#              n_su:=d_geom_param[k]+i*d_geom_param[k+1]
#                                          =refractive index of substrate
#                                           material resp. material
#                                           of adjacent lower coating
#                                           strip
#                                           (Re n_su>0, Im n_su=0)
#
#
#
```

```
#                       +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Switch from external to internal parameter happens
#              automatically if one of the parameters d_geom_param[5*k-2],
#              k=1,...,i_geom_param[2]-2 is larger than one.
#              In this case the upper and lower bounds for the internal
#              parameters d_geom_param[l] with l=5*k-3, l=5*i_geom_param[2]-9
#              are set to 0.9 and 0.1, respectively. The upper and lower
#              bounds for the internal parameters d_geom_param[l] with
#              l=5*k-2, l=5*i_geom_param[2]-8 are set to 0.9 and 0.,
#              respectively.
#
##################
#
# CLASS 5 (i_geom_param[1]=5): input grating to be extended
# ----------------------------------------------------------
#
#       - Grating of input file 'name.inp' extended by a new polygonal
#         interface curve and with refractive indices included into the
#         set of optimization parameters,
#         new polygonal interface:
#              - The new interface connects two given interface points
#                P1 and P2.
#              - These two points are located at the boundary of a
#                domain of a fixed material and are grid points of
#                the file 'name.inp'.
#              - This domain is convex and the periodic boundary lines
#                {(0,y): y real} and  {(dperiod,y): y real} must not
#                intersect its interior.
#              - The new interface curve divides this domain into two.
#              - This new polygonal interface curve is a polygonal
#                function over the straight line segment connecting
#                the two points. In other words, if U is the convex
#                domain fixed by a given material index and if P1 and P2
#                are two given grid points at its boundary, then the
#                j-th corner (j=1,...,m) of the polygonal curve connecting
#                P1 and P2 is chosen at the straight line segment
#
#                   { P = [P1+(P2-P1)*j/(m+1)] + t*n in U:
#                         i)   dl_geom_param[j]<t<du_geom_param[j]
#                         ii)  P+epsilon*n in U
#                         iii) P-epsilon*n in U            },
#
#                         epsilon :=
#                         d_geom_param[2*i_geom_param[2]+i_geom_param[3]+1]
#
#                where n is the unit vector normal at the segment [P1,P2]
#                pointing to the left side of [P1,P2].
#       - Number of string parameters:  ns_geom_param=1
#              &s_geom_param[(1-1)*buffer_size] - 'name.inp'
#                                name of input file for DIPOG-2.1
#                                carrying the geometry information
#                              without the extension by the new
#                              polygonal interface curve
#       - Number of integer parameters: ni_geom_param=6
#               i_geom_param[1]=5
#               i_geom_param[2]=number of materials
#                              (should be number of materials in input
#                               file 'name.inp' plus one if i_geom_param[3]
#                               is positive and should be number of materials
#                               in input file 'name.inp' if i_geom_param[3]
#                               zero)
#               i_geom_param[3]=number of interior knots in the
#                              polygonal interface
#                              (should be non-negative)
#                              If this is zero, then the convex
#                              domain is not divided into two,
#                              and only the refractive indices
#                              are optimized.
#               i_geom_param[4]=index of the grid and interface point P1
#                              (dummy if i_geom_param[3]=0)
#               i_geom_param[5]=index of the grid and interface point P2
#                              (dummy if i_geom_param[3]=0)
#               i_geom_param[6]=index of convex domain which is
#                              divided by the new polygonal interface
#                              (should be between 2 and i_geom_param[2]-1
#                               and is a dummy if i_geom_param[3]=0)
#                              After dividing the convex domain, the
#                              first subdomain (right of the polygonal curve
#                              running from interface point P1 to interface
#                              point P2) inherits the material index
#                              i_geom_param[6] and the index of the second
#                              is appointed to i_geom_param[2]-1.
#                              The domain adjacent to the lower boundary
#                              with  material index i_geom_param[2]-1
#                              before the subdivision now gets the
#                              material index i_geom_param[2].
#       - Number of real parameters:    nd_geom_param=2*i_geom_param[2]+
#                                                    i_geom_param[3]+1
#               k=1,...,i_geom_param[2]
#               d_geom_param[2*k-1]=real part of refractive index of
#                                  grating material with index k
#               d_geom_param[2*k]  =imaginary part of refractive index of
#                                  grating material with index k
#               k=2*i_geom_param[2]+j, j=1,...,i_geom_param[3]
#               d_geom_param[k]    =height of j-th corner of new polygonal
#                                  interface curve over line through the
#                                  given interface points P1 and P2, which
#                                  are located at boundary of the convex
#                                  domain with material index i_geom_param[4]
#                                  and which are the left and right end
#                                  points of the new polygonal interface
#                                  curve
#                                  in other words, the j-th corner point
#                                  is  P=P1+(P2-P1)*j/(m+1)+t*n, where n
#                                  is the vector normal at the segment
#                                  [P1,P2] and where t=d_geom_param[k]
#                                  is a positive or negative real
#                                  (in micro meter,
#                                   -1000<d_geom_param[k]<1000)
#              d_geom_param[2*i_geom_param[2]+i_geom_param[3]+1]
#                                 =threshold for distance of interface
#                                  corner point to boundary of convex
#                                  domain U which is to be split,
#                                  distance is measured in direction of
#                                  normal to [P1,P2],
#                                  this is a dummy if i_geom_param[3]=0,
#                                  (in micro meter,
#                                   0<d_geom_param[2*i_geom_param[2]
#                                     +i_geom_param[3]+1]<1000)
#
#       - Following parameters must be fixed by setting upper bound=
#         lower bound:
#              d_geom_param[2*i_geom_param[2]+i_geom_param[3]+1]
#              d_geom_param[k],k=1,2
#                           i.e. the refractive indices of the cover
#                           material
#              d_geom_param[k],k=2*i_geom_param[2]-1,2*i_geom_param[2]
#                           i.e. the refractive indices of the
#                           substrate material
#
##################
#
# CLASS 6 (i_geom_param[1]=6): EUV bridge
# --------------------------------------
#
#       - Stack of several trapezoids (bridge) in grating
#         with refractive indices included into the set of
#         optimization parameters
#       - Non-stop layers of different heights beneath the stack
#         with refractive indices included into the set of
#         optimization parameters
#         (In other words: Some of the lower additional layers
#         can be added through the set of optimization
#         parameters. Fixed further layers can be added
#         in the GRATING+ILLUMINATION part of this input
#         file.)
#       - Extra layer beside stack, height (>=zero, If upper
#         line of this layers contains a corner of the trapezoids
#         in the stack, then the height of the extra layer and all
#         trapezoid heights of trapezoid beneath this line must
#         be fixed by setting upper bound equal to lower bound.)
#         and refractive indices included into the set of
#         optimization parameters
#       - Number of trapezoids and layers is prescribed
#       - Whole stack (bridge) in one period of the grating
#       - sidewall angles can be restricted by penalty terms
#       - Lower side of stack (bridge) is fixed by two parameters:
#           param_1 = posb/d    = ratio of distance of right lower corner
#                                 from the left boundary line of the period
#                                 and period d
#           param_2 = posa/posb = ratio of distance of left lower corner
#                                 from the left boundary line of the period
#                                 and distance of right lower corner
#                                 from the left boundary line of the period
#       - Each trapezoid is determined by its hight, by the lower side
#         which is the upper side of the adjacent lower trapezoid, and
#         by the upper side prescribed by the two parameters:
#           param_k_1 = b_k/d   = ratio of distance of right upper corner
#                                 from the left boundary line of the period
#                                 and period d
#           param_k_2 = a_k/b_k = ratio of distance of left upper corner
#                                 from the left boundary line of the period
#                                 and distance of right upper corner
#                                 from the left boundary line of the period
#       - Refractive index of the material of each trapezoid
#         and layer is prescribed as an optimization parameter
#       - Number of string parameters:  ns_geom_param=0
#       - Number of integer parameters: ni_geom_param=5
#              i_geom_param[1]=6 (indicator of EUV bridge)
#              i_geom_param[2]=number of different materials
#                             =number of trapezoids in stack (bridge) +
#                              number of non-stop layers beneath stack +
#                              (so far no non-stop layers are allowed)
#                              3 if height of additional layer > 0
#              i_geom_param[2]=number of different materials
#                             =number of trapezoids in stack (bridge) +
#                              number of non-stop layers beneath stack +
#                              (so far no non-stop layers are allowed)
#                              2 if height of additional layer = 0
#              i_geom_param[3]=number of trapezoids in stack
#              i_geom_param[4]=number of non-stop layers beneath stack
#              i_geom_param[5]=index of trapezoid in stack through which
#                              the upper line of the extra layer beside
#                              the stack goes
#                              (1<=i_geom_param[5]<=i_geom_param[3])
#       - Number of real parameters:
#              nd_geom_param=5*i_geom_param[3]+3*i_geom_param[4]+12
#
#
#                  +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Refractive indices:
#
#                                         n_co
#
#                            +---------------------------+
#                           /            n_3              \
#                          /                               \
#                         +---------------------------------+
#                          \                               /
#                           \                             /
#                            \           n_2            /
#                             \                        /
#              +---------------+                      +-----------------+
#                               \                    /
#                 nel            +------------------+          nel
#                               /        n_1         \
#              +---------------+----------------------+-----------------+
#                                        nl_1
#              +--------------------------------------------------------+
#                                        nl_2
#              +--------------------------------------------------------+
#                                        nl_3
#              +--------------------------------------------------------+
#
#                                        n_su
#
#              n_co:=d_geom_param[nd_geom_param-6]+
#                    i*d_geom_param[nd_geom_param-5]
#                   =refractive index of cover material
#                    (Re n_co>0, Im n_co>=0)
#              n_k :=d_geom_param[5*k-1]+i*d_geom_param[5*k]
#                   =refractive index of k-th trapezoid in stack
#                    (Re n_k>0, Im n_k>=0), k=1,...,i_geom_param[3]
#              nel :=d_geom_param[nd_geom_param-8]+
#                    i*d_geom_param[nd_geom_param-7]
#                   =refractive index of extra layer material
#                    (Re nel>0, Im nel>=0)
#              nl_k:=d_geom_param[5*i_geom_param[3]+1+3*k]+
#                    i*d_geom_param[5*i_geom_param[3]+2+3*k]
#                   =refractive index of k-th layer beneath stack
#                    (Re nl_k>0, Im nl_k>=0), k=1,...,i_geom_param[4]
#              n_su:=d_geom_param[nd_geom_param-4]+
#                    i*d_geom_param[nd_geom_param-3]
#                   =refractive index of substrate material resp. material
#                    of adjacent lower coatingstrip
#                    (Re n_su>0, Im n_su=0)
#
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Geometry parameters:
#
#                                    b_k
#              <...........................................>
#              .                                           .
#              .   a_k                                     .
#              <..........>                                .
#              .          .                                .
#              .          .                                .
#              .          .  +--------------------------+  .
#              .          . /                            \ .
#              .          ./                              \.
#              .          +--------------------------------+   ^
#              .           \                              /    :
#       ^      +------------+                            +--------------+
#       :      .             \      k-th trapez.        /      : h_k
#       :      .              \       (k=2)            /       :
#       : hel  .               \                      /        :
#       :      .                \                    /         :
#       v      .  .  .  .  .  .  +------------------+          v
#              .                /                    \
#              +---------------+----------------------+-----------------+
#            (0,0)        (posa,0)               (posb,0)           (d,0)
#       ^      +--------------------------------------------------------+
#       : hl_k                    k-th layer (k=2)
#       v      +--------------------------------------------------------+
#
#              +--------------------------------------------------------+
#
#                  The height hel of the extra layer beside the
#                bridge may be any non-negative number, h_1<hel<h_2
#                                is not supposed!
#
#
#              h_k:=d_geom_param[5*k-4]    =height of k-th trapezoid in stack
#                                           (in micro meter, h_k>0)
#              b_k/d:=d_geom_param[5*k-3]  =ratio of distance of right upper
#                                           corner from the left boundary
#                                           line of the period and period d
#                                           (0<b_k/d<=1)
#              a_k/b_k:=d_geom_param[5*k-2]=ratio of distance of left upper
#                                           corner from the left boundary
#                                           line of the period and distance
#                                           of right upper corner from the
#                                           left boundary line of the period
#                                           (0<=a_k/b_k<1.
#                                            If not both of the parameters
#                                            b_k/d and a_k/b_k are fixed,
#                                            then we require:
#                                            a_k/b_k>0 for b_k/d=1)
#                           k=1,...,i_geom_param[3]
#              posb/d:=d_geom_param[5*i_geom_param[3]+1]
#                                          =ratio of distance of right lower
#                                           corner from the left boundary
#                                           line of the period and period d
#                                           (0<posb/d<=1)
#              posa/posb:=d_geom_param[5*i_geom_param[3]+2]
#                                          =ratio of distance of left lower
#                                           corner from the left boundary
#                                           line of the period and distance
#                                           of right lower corner from the
#                                           left boundary line of the period
#                                           (0<=posa/posb<1.
#                                            If not both of the parameters
#                                            posb/d and posa/posb are fixed,
#                                            then we require:
```

```
#                                    posa/posb>0 for posb/d=1)
#              hel/h_m:=d_geom_param[nd_geom_param-9], m=i_geom_param[5]
#                                  =height of extra layer beside
#                                   stack (bridge) over the
#                                   lower line of the m-th trapezoid
#                                   relative to height of m-th
#                                   trapezoid
#                                   (in micro meter, 0<=hel<=1),
#              hl_k:=d_geom_param[5*i_geom_param[3]+3*k]
#                                  =height of k-th layer beneath stack
#                                   (in micro meter, hl_k>0),
#                           k=1,...,i_geom_param[4],
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Penalty term parameters:
#
#              phi_min=d_geom_param[nd_geom_param-2] minimal angle
#                      for sidewall angles of trapezoids
#                      (except the uppermost trapezoid)
#              phi_max=d_geom_param[nd_geom_param-1] maximal angle
#                      for sidewall angles of trapezoids
#                      (except the uppermost trapezoid)
#              phi_fac=d_geom_param[nd_geom_param] factor of penalty
#                      term
#
#                      In other words, to exclude solutions with too large
#                      or too small sidewall angles phi, a penalty term of
#                      the following form is added to the objective
#                      functional for each sidewall angle phi.
#
#                                 /
#                                |                         2
#                      phi_fac * <   max { 0 , phi-phi_max }
#                                |
#                                 \
#                                                             \
#                                                          2  |
#                                  + max { 0 , phi_min-phi }    >
#                                                             |
#                                                             /
#
#       - Following parameters must be fixed by setting upper bound=
#         lower bound:
#
#           -  d_geom_param[k],k=nd_geom_param-6,...,nd_geom_param
#                   i.e. refractive index of cover and substrate material
#                        and the constants of the penalty terms
#           -  if d_geom_param[nd_geom_param-9]=hel/h_m=0., then:
#              d_geom_param[nd_geom_param-9]
#                   i.e. degenerated height hel
#              d_geom_param[5*k-4], k=1,2,..,i_geom_param[5]-1
#                    i.e. heights of trapezoids beneath upper
#                         boundary line of extra layer
#            -  if d_geom_param[nd_geom_param-9]=hel/h_m=1., then:
#               d_geom_param[nd_geom_param-9]
#                    i.e. degenerated height hel
#               d_geom_param[5*k-4], k=1,2,..,i_geom_param[5]
#                    i.e. heights of trapezoids beneath upper
#                         boundary line of extra layer
#
##################
#
# CLASS 6 (i_geom_param[1]=6): EUV bridge with environment
# ----------------------------------------------------------
#           variable EUV_SWA_90 set to yes
#           ------------------------------
#
#       - Stack of several trapezoids (bridge) in grating
#         with sidewall angles less or equal to 90 degrees
#         and with refractive indices included into the set of
#         optimization parameters
#       - Non-stop layers of different heights beneath the stack
#         with refractive indices included into the set of
#         optimization parameters
#         (In other words: Some of the lower additional layers
#         can be added through the set of optimization
#         parameters. Fixed further layers can be added
#         in the GRATING+ILLUMINATION part of this input
#         file.)
#       - Extra layer beside stack, height (>=zero, If upper
#         line of this layers contains a corner of the trapezoids
#         in the stack, then the height of the extra layer and all
#         trapezoid heights of trapezoid beneath this line must
#         be fixed by setting upper bound equal to lower bound.)
#         and refractive indices included into the set of
#         optimization parameters
#       - Number of trapezoids and layers is prescribed
#       - Whole stack (bridge) in one period of the grating
#       - sidewall angles can be restricted by penalty terms
#       - Lower side of stack (bridge) is fixed by two parameters:
#           param_1 = posb/d    = ratio of distance of right lower corner
#                                 from the left boundary line of the period
#                                 and period d
#           param_2 = posa/posb = ratio of distance of left lower corner
#                                 from the left boundary line of the period
#                                 and distance of right lower corner
#                                 from the left boundary line of the period
#       - Each trapezoid is determined by its hight, by the lower side
#         which is the upper side of the adjacent lower trapezoid, and
#         by the upper side prescribed by the two parameters:
#           param_k_1 = b_k/b_(k-1)
#                               = ratio of distance of right upper corner
#                                 from the left boundary line of the period
#                                 and distance of right lower corner
#                                 from the left boundary line of the period
#                                 b_0:=posb
#           param_k_2 = (a_k-a_(k-1))/(b_k-a_(k-1))
#                               = ratio of difference of x-coordinates of
#                                 left upper and left lower trapezoid corner
#                                 over difference of x-coordinates of
#                                 right upper and left lower trapezoid corner
#                                 a_0:=posa
#       - Refractive index of the material of each trapezoid
#         and layer is prescribed as an optimization parameter
#       - Number of string parameters:  ns_geom_param=0
#       - Number of integer parameters: ni_geom_param=5
#              i_geom_param[1]=6 (indicator of EUV bridge)
#              i_geom_param[2]=number of different materials
#                             =number of trapezoids in stack (bridge) +
#                              number of non-stop layers beneath stack +
#                              (so far no non-stop layers are allowed)
#                              3 if height of additional layer > 0
#              i_geom_param[2]=number of different materials
#                             =number of trapezoids in stack (bridge) +
#                              number of non-stop layers beneath stack +
#                              (so far no non-stop layers are allowed)
#                              2 if height of additional layer = 0
#              i_geom_param[3]=number of trapezoids in stack
#              i_geom_param[4]=number of non-stop layers beneath stack
#              i_geom_param[5]=index of trapezoid in stack through which
#                              the upper line of the extra layer beside
#                              the stack goes
#                              (1<=i_geom_param[5]<=i_geom_param[3])
#       - Number of real parameters:
#              nd_geom_param=5*i_geom_param[3]+3*i_geom_param[4]+12
#
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Refractive indices:
#
#                                        n_co
#
#                            +--------------------------+
#                           /            n_3             \
#                          /                              \
#                         +--------------------------------+
#                          \                              /
#                           \                            /
#                            \           n_2            /
#                             \                        /
#              +---------------+                      +-----------------+
#                               \                    /
#                 nel            +------------------+          nel
#                               /        n_1         \
#              +---------------+----------------------+-----------------+
#                                        nl_1
#              +--------------------------------------------------------+
#                                        nl_2
#              +--------------------------------------------------------+
#                                        nl_3
#              +--------------------------------------------------------+
#
#                                        n_su
#
#              n_co:=d_geom_param[nd_geom_param-6]+
#                    i*d_geom_param[nd_geom_param-5]
#                   =refractive index of cover material
#                    (Re n_co>0, Im n_co>=0)
#              n_k :=d_geom_param[5*k-1]+i*d_geom_param[5*k]
#                   =refractive index of k-th trapezoid in stack
#                    (Re n_k>0, Im n_k>=0), k=1,...,i_geom_param[3]
#              nel :=d_geom_param[nd_geom_param-8]+
#                    i*d_geom_param[nd_geom_param-7]
#                   =refractive index of extra layer material
#                    (Re nel>0, Im nel>=0)
#              nl_k:=d_geom_param[5*i_geom_param[3]+1+3*k]+
#                    i*d_geom_param[5*i_geom_param[3]+2+3*k]
#                   =refractive index of k-th layer beneath stack
#                    (Re nl_k>0, Im nl_k>=0), k=1,...,i_geom_param[4]
#              n_su:=d_geom_param[nd_geom_param-4]+
#                    i*d_geom_param[nd_geom_param-3]
#                   =refractive index of substrate material resp. material
#                    of adjacent lower coatingstrip
#                    (Re n_su>0, Im n_su=0)
#
#
#                      +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#              Geometry parameters:
#
#                                    b_k
#              <...........................................>
#              .                                           .
#              .   a_k                                     .
#              <..........>                                .
#              .          .                                .
#              .          .                                .
#              .          .  +--------------------------+  .
#              .          . /                            \ .
#              .          ./                              \.
#              .          +--------------------------------+   ^
#              .           \                              /    :
#       ^      +------------+                            +--------------+
#       :      .             \       k-th trapez.       /      : h_k
#       :      .              \        (k=2)           /       :
#       : hel  .               \                      /        :
#       :      .                \                    /         :
#       v      .  .  .  .  .  .  +------------------+          v
#              .                /                    \
#              +---------------+----------------------+-----------------+
#            (0,0)        (posa,0)               (posb,0)             (d,0)
#       ^      +--------------------------------------------------------+
#       : hl_k                     k-th layer (k=2)
#       v      +--------------------------------------------------------+
#
#              +--------------------------------------------------------+
#
#                  The height hel of the extra layer beside the
#                bridge may be any non-negative number, h_1<hel<h_2
#                                is not supposed!
#
#
#              h_k:=d_geom_param[5*k-4]    =height of k-th trapezoid in stack
#                                           (in micro meter, h_k>0)
#              b_k/b_(k-1):=d_geom_param[5*k-3]
#                                          =ratio of distance of right upper
#                                           corner from the left boundary
#                                           line of the period and
#                                           distance of right lower
#                                           corner from the left boundary
#                                           line of the period
#                                           (0<b_k/b_(k-1)<=1, b_0:=posb)
#              (a_k-a_(k-1))/(b_k-a_(k-1)):=d_geom_param[5*k-2]
#                                          =ratio of difference of
#                                           x-coordinates of left upper and
#                                           left lower trapezoid corner
#                                           over difference of x-coordinates
#                                           of right upper and left lower
#                                           trapezoid corner
#                                           (0<=(a_k-a_(k-1))/(b_k-a_(k-1))<1,
#                                            a_0:=posa)
#                           k=1,...,i_geom_param[3]
#              posb/d:=d_geom_param[5*i_geom_param[3]+1]
#                                          =ratio of distance of right lower
#                                           corner from the left boundary
#                                           line of the period and period d
#                                           (0<posb/d<=1)
#              posa/posb:=d_geom_param[5*i_geom_param[3]+2]
#                                          =ratio of distance of left lower
#                                           corner from the left boundary
#                                           line of the period and distance
#                                           of right lower corner from the
#                                           left boundary line of the period
#                                           (0<=posa/posb<1.
#                                            If not both of the parameters
#                                            posb/d and posa/posb are fixed,
#                                            then we require:
#                                            posa/posb>0 for posb/d=1)
#              hel/h_m:=d_geom_param[nd_geom_param-9], m=i_geom_param[5]
#                                          =height of extra layer beside
#                                           stack (bridge) over the
#                                           lower line of the m-th trapezoid
#                                       relative to height of m-th
#                                       trapezoid
#                                       (in micro meter, 0<=hel<=1),
#             hl_k:=d_geom_param[5*i_geom_param[3]+3*k]
#                                      =height of k-th layer beneath stack
#                                       (in micro meter, hl_k>0),
#                           k=1,...,i_geom_param[4],
#
#                     +  +  +  +  +  +  +  +  +  +  +  +  +
#
#
#
#             Penalty term parameters:
#
#             phi_min=d_geom_param[nd_geom_param-2] minimal angle
#                     for sidewall angles of trapezoids
#                     (except the uppermost trapezoid)
#             phi_max=d_geom_param[nd_geom_param-1] maximal angle
#                     for sidewall angles of trapezoids
#                     (except the uppermost trapezoid)
#             phi_fac=d_geom_param[nd_geom_param] factor of penalty
#                     term
#
#                     In other words, to exclude solutions with too large
#                     or too small sidewall angles phi, a penalty term of
#                     the following form is added to the objective
#                     functional for each sidewall angle phi.
#
#                                /
#                               |                         2
#                     phi_fac * <   max { 0 , phi-phi_max }
#                               |
#                                \
#                                                             \
#                                                          2  |
#                                  + max { 0 , phi_min-phi }    >
#                                                             |
#                                                             /
#
#      - Following parameters must be fixed by setting upper bound=
#        lower bound:
#
#          -  d_geom_param[k],k=nd_geom_param-6,...,nd_geom_param
#                  i.e. refractive index of cover and substrate material
#                       and the constants of the penalty terms
#          -  if d_geom_param[nd_geom_param-9]=hel/h_m=0., then:
#             d_geom_param[nd_geom_param-9]
#                  i.e. degenerated height hel
#             d_geom_param[5*k-4], k=1,2,..,i_geom_param[5]-1
#                  i.e. heights of trapezoids beneath upper
#                       boundary line of extra layer
#          -  if d_geom_param[nd_geom_param-9]=hel/h_m=1., then:
#             d_geom_param[nd_geom_param-9]
#                  i.e. degenerated height hel
#              d_geom_param[5*k-4], k=1,2,..,i_geom_param[5]
#                   i.e. heights of trapezoids beneath upper
#                        boundary line of extra layer
#
##################
# Number nd_geom_param of real parameters:
  6
# Lower bounds dl_geom_param (nd_geom_param numbers in nd_geom_param lines):
  -0.5
  -0.5
  1.
  0.
  1.5
  0.
# Upper bounds du_geom_param (nd_geom_param numbers in nd_geom_param lines):
  0.5
  0.5
  1.
  0.
  1.5
  0.
# Number ni_geom_param of integer parameters:
  3
# Integer params. i_geom_param (ni_geom_param numbers in ni_geom_param lines):
  1
  2
  2
# Number ns_geom_param of name parameters:
  0
# Parameter names s_geom_param.
# Each in one line, i.e., ns_geom_param lines.
# Parameters:
######################################################################
# Parameters d_geom_param of initial grating.
# d_geom_param (nd_geom_param numbers in nd_geom_param lines).
# With dl_geom_param[i]<=d_geom_param[i]<=du_geom_param[i] for all i.
# (If initial solution is to be sought by a deterministic search
#  algorithm, then add the line: ``  no n_0 n_1 n_2'', where n_0
#  is the refinement level for the FEM computation, n_1 is the
#  number of maximal subdivision points per dimension, and n_2 is
#  an indicator. If n_2=1, then the minimum is improved by computing
#  the local minimum of the linear Taylor polynomial around each
#  mesh point.)
# Parameters:
  0.07
  -0.04
  1.
  0.
  1.5
  0.
######################################################################
#                                                                    #
#          L E V E L   O F   D I S C R E T I Z A T I O N             #
#                                                                    #
```

```
######################################################################
# Number of levels (Lev).
#    Computation is performed on this level.
#    Alternatively, an incremental input is possible.
#    E.g.
#      ``  I 2 8 3''.
#      The last means that computation is to be done for
#      the levels ``2+i*3'' with i=0,1,2,... as long as
#      level ``2+i*3'' is less or equal to ``8''.
#      The initial solution of the computation on level ``2''
#      is the input initial solution. The initial solution
#      of the computation on level ``2+i*3'' for i>0 is the
#      final solution of the level ``2+(i-1)*3''.
# Number:
  1
######################################################################
#                                                                    #
#             O B J E C T I V E   F U N C T I O N A L                #
#                                                                    #
######################################################################
#
#                                  reflected
#   value =    w_lin_ene_re * energy             +
#
#
#                                  transmitted
#              w_lin_ene_tr * energy             +
#
#
#                                  total
#              w_lin_ene_to * energy             +
#
#
#    n_lin_re
#      ---                         reflected
#       >      w_lin_re * efficiency             +
#      ---             j           o_lin_re
#      j=1                                 j
#
#    n_lin_tr
#      ---                          transmitted
#       >      w_lin_tr  * efficiency             +
#      ---             j            o_lin_tr
#      j=1                                  j
#
#    n_qua_re
#      ---                          reflected            2
#       >      w_qua_re  [ efficiency          - c_qua_re ]     +
#      ---             j            o_qua_re            j
#      j=1                                  j
#
#    n_qua_tr
#      ---                          transmitted            2
#       >      w_qua_tr  [ efficiency            - c_qua_tr ]   +
#      ---             j             o_qua_tr              j
#      j=1                                    j
#
#    n_lin_1_re
#      ---                           reflected
#       >      w_lin_1_re * efficiency_1            +
#      ---               j             o_lin_1_re
#      j=1                                       j
#
#    n_lin_1_tr
#      ---                            transmitted
#       >      w_lin_1_tr  * efficiency_1             +
#      ---               j               o_lin_1_tr
#      j=1                                        j
#
#    n_qua_1_re
#      ---                          reflected                2
#       >      w_qua_1_re  [ efficiency            - c_qua_1_re ]     +
#      ---             j              o_qua_1_re             j
#      j=1                                      j
#
#    n_qua_1_tr
#      ---                            transmitted              2
#       >      w_qua_1_tr  [ efficiency_1            - c_qua_1_tr ]   +
#      ---             j                o_qua_1_tr             j
#      j=1                                        j
#
#    n_lin_2_re
#      ---                           reflected
#       >      w_lin_2_re * efficiency_2            +
#      ---               j             o_lin_2_re
#      j=1                                       j
#
#    n_lin_2_tr
#      ---                            transmitted
#       >      w_lin_2_tr  * efficiency_2             +
#      ---               j               o_lin_2_tr
#      j=1                                        j
#
#    n_qua_2_re
#      ---                            reflected                2
#       >      w_qua_2_re  [ efficiency_2            - c_qua_2_re ]     +
#      ---               j              o_qua_2_re             j
#      j=1                                        j
#
#    n_qua_2_tr
#      ---                            transmitted              2
#       >      w_qua_2_tr  [ efficiency_2            - c_qua_2_tr ]   +
#      ---               j              o_qua_2_tr             j
#      j=1                                        j
#
#    n_psh_1_re
#      ---                            reflected                2
#       >      w_psh_1_re  [ phaseshift_1            - c_psh_1_re ]     +
#      ---                 j                o_psh_1_re              j *
#      j=1                                            j
#
#    n_psh_1_tr
#      ---                                  transmitted               2
#       >      w_psh_1_tr  [ phaseshift_1               - c_psh_1_tr ]   +
#      ---                 j                o_psh_1_tr              j *
#      j=1                                            j
#
#    n_psh_2_re
#      ---                                  reflected                2
#       >      w_psh_2_re  [ phaseshift_2             - c_psh_2_re ]     +
#      ---                 j                o_psh_2_re             j *
#      j=1                                            j
#
#    n_psh_2_tr
#      ---                                  transmitted               2
#       >      w_psh_2_tr  [ phaseshift_2               - c_psh_2_tr ]    +
#      ---                 j                o_psh_2_tr              j *
#      j=1                                            j
#
#                         reflected            2
#    w_qua_ene_re [ energy          - c_ene_re ]    +
#
#                         transmitted            2
#    w_qua_ene_tr [ energy            - c_ene_tr ]  +
#
#                         total            2
#    w_qua_ene_to [ energy      - c_ene_to ]
#
#                                       2       2
#                          Here: [ p-c ]  := sin ( Pi*(p-c)/360 )
#                                       *
#
########################
#                                                      refl./transm.
#      Note that the angles of phase shift phaseshift_m
#      are between -180 and 180.                       o
#
#
#                                             refl./transm.
#      Note that the efficiencies efficiency_1              denote the
#                                             o
#      the efficiency TE-part for TE/TM output resp. the S-part for
#      Jones output.
#
#                                             refl./transm.
#      Note that the efficiencies efficiency_2              denote the
#                                             o
#      the efficiency TM-part for TE/TM output resp. the P-part for
#      Jones output.
#
#      For 3.Comp output these first and second efficiency terms are
#      not allowed, i.e. the corresponding n_lin/qua_m_re/tr are
#      to be set to zero.
#
########################
#
#      If the wavelength or/and the incidence angles or/and the
#      polarization run over a fixed set of values, then the last value
#      of the objective function is replaced by the sum over the last
#      values depending on the wavelengths or/and the angles or/and the
#      polarization. In this case, the values w_... equal to
#
#
#           w_lin_ene_re,   w_lin_ene_tr,   w_lin_ene_to,
#
#
#           w_lin_re ,      w_lin_tr ,
#                   j               j
#
#           w_lin_k_re ,    w_lin_k_tr , k=1,2
#                     j               j
#
#           w_qua_re ,      w_qua_tr ,
#                   j               j
#
#           w_qua_k_re ,    w_qua_k_tr , k=1,2
#                     j               j
#
#           w_psh_k_re ,    w_psh_k_tr , k=1,2
#                     j               j
#
#           w_qua_ene_re,   w_qua_ene_tr,   w_qua_ene_to
#
#
#      and, similarly, the corresponding prescribed parameters c_...
#      equal to
#
#
#           c_qua_re ,      c_qua_tr ,
#                   j               j
#
#           c_qua_k_re ,    c_qua_k_tr , k=1,2
#                     j               j
#
#           c_psh_k_re ,    c_psh_k_tr , k=1,2
#                     j               j
#
#           c_ene_re,       c_ene_tr,    c_ene_to
#
#
#      may depend on the wavelengths or/and the angles or/and the
#      polarization type. If the n_...  values of c_... depend on
#      m wavelengths resp. angles theta resp. angles phi, then
#      the input consists of m*n_...+1 lines. The first contains
#      the indicator ‘‘WAL’’ resp. ‘‘ATH’’ resp. ‘‘APH’’ and is
#      followed by m*n_... lines each containing a number c_...
#       for one value of the wavelength resp. angle theta
#       resp. angle phi and for one order of mode. If the n_... values
#       of c_... depends on m1 wavelengths + m2 angles theta resp. m1
#       wavelengths + m2 angles phi resp. m1 angles theta + m2 angles
#       phi, then the input consists of m1*m2*n_...+1 lines. The first
#       contains the indicator ``W+T'' resp. ``W+P'' resp. ``T+P'' and
#       is followed by m1*m2*n_... lines each containing a number
#       c_... for one pair of values wavelength + angle theta resp.
#       wavelength + angles phi resp. angle theta + angle phi and for
#       one order of mode. If the n_... values of c_... depend on
#       m1 wavelengths + m2 angles theta + m3 angles phi, then the input
#       consists of m1*m2*m3*n_...+1 lines. The first contains the
#       indicator ``WTP'' and is followed by m1*m2*m2*n_... lines each
#       containing a number c_... for one triple of values wavelength
#       + angle theta + angle phi. E.g.:
#
#
#        ``   WAL
#             c_...(lambda1,mode1    )
#             c_...(lambda1,mode2    )
#               ...
#             c_...(lambda1,moden_...)
#             c_...(lambda2,mode1    )
#             c_...(lambda2,mode2    )
#               ...
#             c_...(lambda2,moden_...)
#                  ...
#             c_...(lambdam,mode1    )
#             c_...(lambdam,mode2    )
#               ...
#             c_...(lambdam,moden_...)  ''
#
#
#         ``  W+P
#             c_...(lambda1 ,phi1 ,mode1    )
#             c_...(lambda1 ,phi1 ,mode2    )
#               ...
#             c_...(lambda1 ,phi1 ,moden_...)
#             c_...(lambda1 ,phi2 ,mode1    )
#             c_...(lambda1 ,phi2 ,mode2    )
#               ...
#             c_...(lambda1 ,phi2 ,moden_...)
#                  ...
#             c_...(lambda1 ,phim2,mode1    )
#             c_...(lambda1 ,phim2,mode2    )
#               ...
#             c_...(lambda1 ,phim2,moden_...)
#             c_...(lambda2 ,phi1 ,mode1    )
#             c_...(lambda2 ,phi1 ,mode2    )
#               ...
#             c_...(lambda2 ,phi1 ,moden_...)
#             c_...(lambda2 ,phi2 ,mode1    )
#             c_...(lambda2 ,phi2 ,mode2    )
#               ...
#            c_...(lambda2 ,phi2 ,moden_...)
#                 ...
#            c_...(lambda2 ,phim2,mode1    )
#            c_...(lambda2 ,phim2,mode2    )
#              ...
#            c_...(lambda2 ,phim2,moden_...)
#                    ...
#            c_...(lambdam1,phi1 ,mode1    )
#            c_...(lambdam1,phi1 ,mode2    )
#              ...
#            c_...(lambdam1,phi1 ,moden_...)
#            c_...(lambdam1,phi2 ,mode1    )
#            c_...(lambdam1,phi2 ,mode2    )
#              ...
#            c_...(lambdam1,phi2 ,moden_...)
#                 ...
#            c_...(lambdam1,phim2,mode1    )
#            c_...(lambdam1,phim2,mode2    )
#              ...
#            c_...(lambdam1,phim2,moden_...)  ''
#
#
#        ``  WTP
#            c_...(lambda1 ,theta1 ,phi1 ,mode1    )
#            c_...(lambda1 ,theta1 ,phi1 ,mode2    )
#              ...
#            c_...(lambda1 ,theta1 ,phi1 ,moden_...)
#            c_...(lambda1 ,theta1 ,phi2 ,mode1    )
#            c_...(lambda1 ,theta1 ,phi2 ,mode2    )
#              ...
#            c_...(lambda1 ,theta1 ,phi2 ,moden_...)
#                 ...
#            c_...(lambda1 ,theta1 ,phim3,mode1    )
#            c_...(lambda1 ,theta1 ,phim3,mode2    )
#              ...
#            c_...(lambda1 ,theta1 ,phim3,moden_...)
#            c_...(lambda1 ,theta2 ,phi1 ,mode1    )
#            c_...(lambda1 ,theta2 ,phi1 ,mode2    )
#              ...
#            c_...(lambda1 ,theta2 ,phi1 ,moden_...)
#            c_...(lambda1 ,theta2 ,phi2 ,mode1    )
#            c_...(lambda1 ,theta2 ,phi2 ,mode2    )
#              ...
#            c_...(lambda1 ,theta2 ,phi2 ,moden_...)
#                 ...
#            c_...(lambda1 ,theta2 ,phim3,mode1    )
#            c_...(lambda1 ,theta2 ,phim3,mode2    )
#              ...
#            c_...(lambda1 ,theta2 ,phim3,moden_...)
#                    ...
#            c_...(lambda1 ,thetam2,phi1 ,mode1    )
#            c_...(lambda1 ,thetam2,phi1 ,mode2    )
#              ...
#            c_...(lambda1 ,thetam2,phi1 ,moden_...)
#              c_...(lambda1 ,thetam2,phi2 ,mode1    )
#              c_...(lambda1 ,thetam2,phi2 ,mode2    )
#                ...
#              c_...(lambda1 ,thetam2,phi2 ,moden_...)
#                   ...
#              c_...(lambda1 ,thetam2,phim3,mode1    )
#              c_...(lambda1 ,thetam2,phim3,mode2    )
#                ...
#              c_...(lambda1 ,thetam2,phim3,moden_...)
#              c_...(lambda2 ,theta1 ,phi1 ,mode1    )
#              c_...(lambda2 ,theta1 ,phi1 ,mode2    )
#                ...
#              c_...(lambda2 ,theta1 ,phi1 ,moden_...)
#              c_...(lambda2 ,theta1 ,phi2 ,mode1    )
#              c_...(lambda2 ,theta1 ,phi2 ,mode2    )
#                ...
#              c_...(lambda2 ,theta1 ,phi2 ,moden_...)
#                   ...
#              c_...(lambda2 ,theta1 ,phim3,mode1    )
#              c_...(lambda2 ,theta1 ,phim3,mode2    )
#                ...
#              c_...(lambda2 ,theta1 ,phim3,moden_...)
#              c_...(lambda2 ,theta2 ,phi1 ,mode1    )
#              c_...(lambda2 ,theta2 ,phi1 ,mode2    )
#                ...
#              c_...(lambda2 ,theta2 ,phi1 ,moden_...)
#              c_...(lambda2 ,theta2 ,phi2 ,mode1    )
#              c_...(lambda2 ,theta2 ,phi2 ,mode2    )
#                ...
#              c_...(lambda2 ,theta2 ,phi2 ,moden_...)
#                   ...
#              c_...(lambda2 ,theta2 ,phim3,mode1    )
#              c_...(lambda2 ,theta2 ,phim3,mode2    )
#                ...
#              c_...(lambda2 ,theta2 ,phim3,moden_...)
#                      ...
#              c_...(lambda2 ,thetam2,phi1 ,mode1    )
#              c_...(lambda2 ,thetam2,phi1 ,mode2    )
#                ...
#              c_...(lambda2 ,thetam2,phi1 ,moden_...)
#              c_...(lambda2 ,thetam2,phi2 ,mode1    )
#              c_...(lambda2 ,thetam2,phi2 ,mode2    )
#                ...
#              c_...(lambda2 ,thetam2,phi2 ,moden_...)
#                   ...
#              c_...(lambda2 ,thetam2,phim3,mode1    )
#              c_...(lambda2 ,thetam2,phim3,mode2    )
#                ...
#              c_...(lambda2 ,thetam2,phim3,moden_...)
#                         ...
#              c_...(lambdam1,theta1 ,phi1 ,mode1    )
#              c_...(lambdam1,theta1 ,phi1 ,mode2    )
#                ...
#              c_...(lambdam1,theta1 ,phi1 ,moden_...)
#           c_...(lambdam1,theta1 ,phi2 ,mode1    )
#           c_...(lambdam1,theta1 ,phi2 ,mode2    )
#             ...
#           c_...(lambdam1,theta1 ,phi2 ,moden_...)
#                ...
#           c_...(lambdam1,theta1 ,phim3,mode1    )
#           c_...(lambdam1,theta1 ,phim3,mode2    )
#             ...
#           c_...(lambdam1,theta1 ,phim3,moden_...)
#           c_...(lambdam1,theta2 ,phi1 ,mode1    )
#           c_...(lambdam1,theta2 ,phi1 ,mode2    )
#             ...
#           c_...(lambdam1,theta2 ,phi1 ,moden_...)
#           c_...(lambdam1,theta2 ,phi2 ,mode1    )
#           c_...(lambdam1,theta2 ,phi2 ,mode2    )
#             ...
#           c_...(lambdam1,theta2 ,phi2 ,moden_...)
#                ...
#           c_...(lambdam1,theta2 ,phim3,mode1    )
#           c_...(lambdam1,theta2 ,phim3,mode2    )
#             ...
#           c_...(lambdam1,theta2 ,phim3,moden_...)
#                   ...
#           c_...(lambdam1,thetam2,phi1 ,mode1    )
#           c_...(lambdam1,thetam2,phi1 ,mode2    )
#             ...
#           c_...(lambdam1,thetam2,phi1 ,moden_...)
#           c_...(lambdam1,thetam2,phi2 ,mode1    )
#           c_...(lambdam1,thetam2,phi2 ,mode2    )
#             ...
#           c_...(lambdam1,thetam2,phi2 ,moden_...)
#                ...
#           c_...(lambdam1,thetam2,phim3,mode1    )
#           c_...(lambdam1,thetam2,phim3,mode2    )
#             ...
#           c_...(lambdam1,thetam2,phim3,moden_...)  ''
#
#       If, additionally, the type of polarization runs over two types
#       (in this case input type of polarization must be TE/TM:
#       first type poltype1=TE and second type poltype2=TM), then the
#       value of the objective function is replaced by the additional
#       sum over the polarization types. In this case, the values
#       c_... may depend on the type of polarization, too. The input of
#       these values is
#
#         ``  POL
#           c_...(poltyp1,mode1)
#           c_...(poltyp1,mode2)
#             ...
#           c_...(poltyp1,moden_...)
#           c_...(poltyp2,mode1)
#           c_...(poltyp2,mode2)
#             ...
#           c_...(poltyp2,moden_...) ''
```

```
#
#      if the values do not depend on lambda, theta, and phi. In case
#      it depends on lambda, theta, or phi, the input starts with
#      ‘‘WAL+POL’’ for ‘‘WAL’’, ‘‘ATH+POL’’ for ‘‘ATH’’, ‘‘APH+POL’’
#      for ‘‘APH’’, ‘‘W+T+POL’’ for ‘‘W+T’’, ‘‘W+P+POL’’ for ‘‘W+P’’,
#      ‘‘T+P+POL’’ for ‘‘T+P’’, ‘‘WTP+POL’’ for ‘‘WTP’’. Then there
#      follows the doubled number of lines with values of c_... .
#      This corresponds to a loop over lambda/theta/phi, as usually.
#      In the inner of the loop, for fixed lambda, theta, and phi,
#      first the n_... values for the first polarization type poltyp1
#      are listed in the usual order. Then the n_... values for the
#      second polarization type poltyp2 follow.
#
#      If a value c_... or a value w_... depends on the wavelengths
#      or/and the angles or/and the polarization type, then the
#      corresponding value w_... or a value c_... must be given in
#      the same way. In other words, if, e.g., w_... is constant
#      but c_... depends on m values of wavelength, then the
#      input for w_... must start with the line ‘‘WAL’’ and after
#      wards the constant value w_... must be repeated m-times in
#      m separate lines.
#
########################
#
#      If the type of polarization and coordinate system for the
#      incoming wave vector is ‘‘TE/TM’’, then the phase shifts
#      are computed first for TE and then for TM, and terms like
#
#
#    n_psh_1_re
#      ---                       reflected,TE/TM             2
#       >      w_psh_1_re  [ ps_1               - c_psh_1_re ]     +
#      ---               j       o_psh_1_re                 j *
#      j=1                                        j
#
#    n_psh_1_tr
#      ---                       transmitted ,TE/TM             2
#       >      w_psh_1_tr  [ ps_1                  - c_psh_1_tr ]   +
#      ---               j       o_psh_1_tr                    j *
#      j=1                                j
#
#    n_psh_2_re
#      ---                       reflected,TE/TM             2
#       >      w_psh_2_re  [ ps_2               - c_psh_2_re ]     +
#      ---               j       o_psh_2_re                 j *
#      j=1                                j
#
#    n_psh_2_tr
#      ---                       transmitted,TE/TM              2
#       >      w_psh_2_tr  [ ps_2                  - c_psh_2_tr ]
#      ---               j       o_psh_2_tr                    j *
#      j=1                                j
#
#
```

```
#
#      with
#
#
#        reflected,TE/TM              reflected,TE              reflected,TM
#    ps_1               = phaseshift_1            - phaseshift_1
#        o_psh_1_re                   o_psh_1_re                o_psh_1_re
#                   j                            j                          j
#
#        transmitted,TE/TM              transmitted,TE            transmitted,TM
#    ps_1                 = phaseshift_1            - phaseshift_1
#        o_psh_1_tr                     o_psh_1_tr                o_psh_1_tr
#                   j                              j                          j
#
#        reflected,TE/TM              reflected,TE              reflected,TM
#    ps_2               = phaseshift_2            - phaseshift_2
#        o_psh_2_re                   o_psh_2_re                o_psh_2_re
#                   j                            j                          j
#
#        transmitted,TE/TM              transmitted,TE            transmitted,TM
#    ps_2                 = phaseshift_2            - phaseshift_2
#        o_psh_2_tr                     o_psh_2_tr                o_psh_2_tr
#                   j                              j                          j
#
#
#      are of interest. To indicate that terms of this type are to be
#      included instead of the terms depending on one phaseshift only
#      add a TE/TM before the input of the numbers n_psh_1_re,
#      n_psh_2_re, n_psh_1_tr, and n_psh_2_tr, respectively. E.g.
#
#                  # n_phs_2_tr:
#                    TE/TM
#                    1
#
#      In such a case an input of weights w_... and prescribed values
#      c_... beginning with ...+POL is wrong since the phaseshifts
#      for different polarization are included into one term, only.
#
#      If the type of polarization and coordinate system for the
#      incoming wave vector is ``TE/TM'', if the type of output
#      is ``TE/TM'', and if the angle phi of illumination is zero
#      (classical case), then the phase shift are computed first
#      for TE and then for TM, and terms like
#
#
#    n_psh_1_re
#      ---                       reflected,TE/TM             2
#       >      w_psh_1_re  [ PS_1               - c_psh_1_re ]     +
#      ---                j      o_psh_1_re                 j *
#      j=1                                         j
#
#    n_psh_1_tr
#      ---                       transmitted ,TE/TM             2
#       >      w_psh_1_tr  [ PS_1                  - c_psh_1_tr ]   +
#      ---               j       o_psh_1_tr                  j *
#      j=1                                j
#
#
#      with
#
#
#        reflected,TE/TM              reflected,TE              reflected,TM
#    PS_1               = phaseshift_1            - phaseshift_2
#        o_psh_1_re                   o_psh_1_re                o_psh_1_re
#                   j                            j                         j
#
#        transmitted,TE/TM              transmitted,TE            transmitted,TM
#    PS_1                 = phaseshift_1            - phaseshift_2
#        o_psh_1_tr                     o_psh_1_tr                o_psh_1_tr
#                   j                              j                         j
#
#      are of interest. To indicate that terms of this type are to be
#      included instead of the terms depending on one phaseshift only
#      add a CL:TE/TM before the input of the numbers n_psh_1_re
#      and n_psh_1_tr, respectively. E.g.
#
#                  # n_phs_1_tr:
#                    CL:TE/TM
#                    2
#
#      In such a case an input of weights w_... and prescribed values
#      c_... beginning with ...+POL is wrong since the phaseshifts
#      for different polarization are included into one term, only.
#
########################
#
#      If the weights for the efficiency terms should be the
#      reciprocal squared uncertainty and if this uncertainty
#      should be a function of the prescribed efficiency value, e.g.:
#
#         w_qua_1_tr=1/(u*u), u=f(c_qua_1_tr), f(E):=sqrt(E*E+1e-2)
#
#      Then the input is as follows:
#
#         ``# n_qua_1_tr
#             2
#           # w_qua_1_tr
#             unc.fct. sqrt(E*E+1e-2)
#             0.
#             1.
#           # o_qua_1_tr
#             -1
#              0
#           # c_qua_1_re
#             10.
#             13.  ''
#
#      Note that the two input values for the w_qua_1_tr
#      are dummy values. The final values will be computed as
#
#        w_qua_1_tr=1/(u*u), u=sqrt(c_qua_1_re*c_qua_1_re+1e-2)
#                            with c_qua_1_re=10 or c_qua_1_re=13.
#
#      if the dummy input for w_qua_1_tr is positive, and it
#      will be set to zero if w_qua_1_tr is less or equal to zero.
#
########################
#
#LINEAR TERMS, REFLECTED EFFICIENCY
# w_ene_lin_re
  0
# w_ene_lin_tr
  0
# w_ene_lin_to
  0
# n_lin_re:
  0
# w_lin_re (n_lin_re numbers in n_lin_re lines):
# o_lin_re (n_lin_re numbers in n_lin_re lines):
#LINEAR TERMS, FIRST (TE or S) REFLECTED EFFICIENCY
# n_lin_1_re:
  0
# w_lin_1_re (n_lin_1_re numbers in n_lin_1_re lines):
# o_lin_1_re (n_lin_1_re numbers in n_lin_1_re lines):
#LINEAR TERMS, SECOND (TM or P) REFLECTED EFFICIENCY
# n_lin_2_re:
  0
# w_lin_2_re (n_lin_2_re numbers in n_lin_2_re lines):
# o_lin_2_re (n_lin_2_re numbers in n_lin_2_re lines):
#LINEAR TERMS, TRANSMITTED EFFICIENCY
# n_lin_tr:
  0
# w_lin_tr (n_lin_tr numbers in n_lin_tr lines):
# o_lin_tr (n_lin_tr numbers in n_lin_tr lines):
#LINEAR TERMS, FIRST (TE or S) TRANSMITTED EFFICIENCY
# n_lin_1_tr:
  0
# w_lin_1_tr (n_lin_1_tr numbers in n_lin_1_tr lines):
# o_lin_1_tr (n_lin_1_tr numbers in n_lin_1_tr lines):
#LINEAR TERMS, SECOND (TM or P) TRANSMITTED EFFICIENCY
# n_lin_2_tr:
  0
# w_lin_2_tr (n_lin_2_tr numbers in n_lin_2_tr lines):
# o_lin_2_tr (n_lin_2_tr numbers in n_lin_2_tr lines):
#QUADRATIC TERMS, REFLECTED EFFICIENCY
# n_qua_re:
  0
# w_qua_re (n_qua_re numbers in n_qua_re lines):
# o_qua_re (n_qua_re numbers in n_qua_re lines):
# c_qua_re (n_qua_re numbers in n_qua_re lines):
#QUADRATIC TERMS, FIRST (TE or S) REFLECTED EFFICIENCY
# n_qua_1_re:
  1
# w_qua_1_re (n_qua_1_re numbers in n_qua_1_re lines):
  0.03
# o_qua_1_re (n_qua_1_re numbers in n_qua_1_re lines):
  0
# c_qua_1_re (n_qua_1_re numbers in n_qua_1_re lines):
  51.603605
#QUADRATIC TERMS, SECOND (TM or P) REFLECTED EFFICIENCY
# n_qua_2_re:
  0
# w_qua_2_re (n_qua_2_re numbers in n_qua_2_re lines):
# o_qua_2_re (n_qua_2_re numbers in n_qua_2_re lines):
# c_qua_2_re (n_qua_2_re numbers in n_qua_2_re lines):
#QUADRATIC TERMS, TRANSMITTED EFFICIENCY
# n_qua_tr:
  0
# w_qua_tr (n_qua_tr numbers in n_qua_tr lines):
# o_qua_tr (n_qua_tr numbers in n_qua_tr lines):
# c_qua_tr (n_qua_tr numbers in n_qua_tr lines):
#QUADRATIC TERMS, FIRST (TE or S) TRANSMITTED EFFICIENCY
# n_qua_1_tr:
  0
# w_qua_1_tr (n_qua_1_tr numbers in n_qua_1_tr lines):
# o_qua_1_tr (n_qua_1_tr numbers in n_qua_1_tr lines):
# c_qua_1_tr (n_qua_1_tr numbers in n_qua_1_tr lines):
#QUADRATIC TERMS, SECOND (TM or P) TRANSMITTED EFFICIENCY
# n_qua_2_tr:
  0
# w_qua_2_tr (n_qua_2_tr numbers in n_qua_2_tr lines):
# o_qua_2_tr (n_qua_2_tr numbers in n_qua_2_tr lines):
# c_qua_2_tr (n_qua_2_tr numbers in n_qua_2_tr lines):
#QUADRATIC TERMS, FIRST (TE or S) REFLECTED PHASE SHIFT
# n_phs_1_re:
  1
# w_phs_1_re (n_phs_1_re numbers in n_phs_1_re lines):
  30.
# o_phs_1_re (n_phs_1_re numbers in n_phs_1_re lines):
  0
# c_phs_1_re (n_phs_1_re numbers in n_phs_1_re lines):
  -91.464440
#QUADRATIC TERMS, FIRST (TE or S) TRANSMITTED PHASE SHIFT
# n_phs_1_tr:
  0
# w_phs_1_tr (n_phs_1_tr numbers in n_phs_1_tr lines):
# o_phs_1_tr (n_phs_1_tr numbers in n_phs_1_tr lines):
# c_phs_1_tr (n_phs_1_tr numbers in n_phs_1_tr lines):
#QUADRATIC TERMS, SECOND (TM or P) REFLECTED PHASE SHIFT
# n_phs_2_re:
  0
# w_phs_2_re (n_phs_2_re numbers in n_phs_2_re lines):
# o_phs_2_re (n_phs_2_re numbers in n_phs_2_re lines):
# c_phs_2_re (n_phs_2_re numbers in n_phs_2_re lines):
#QUADRATIC TERMS, SECOND (TM or P) TRANSMITTED PHASE SHIFT
# n_phs_2_tr:
  0
# w_phs_2_tr (n_phs_2_tr numbers in n_phs_2_tr lines):
# o_phs_2_tr (n_phs_2_tr numbers in n_phs_2_tr lines):
# c_phs_2_tr (n_phs_2_tr numbers in n_phs_2_tr lines):
#QUADRATIC TERMS, REFLECTED ENERGY
# w_qua_ene_re:
  0.
# c_ene_re (no line if w_ene_re=0.):
#QUADRATIC TERMS, TRANSMITTED ENERGY
# w_qua_ene_tr:
  0.
# c_ene_tr (no line if w_ene_tr=0.):
#QUADRATIC TERMS, TOTAL ENERGY
# w_qua_ene_to:
  0.
# c_ene_to (no line if w_ene_to=0.):
######################################################################
#                                                                    #
#          O P T I M I Z A T I O N   A L G O R I T H M               #
#                                                                    #
######################################################################
# Data for optimization algorithm.
#
#   - maximal number of iterations
#     This is usually a positive number.
#     However, if the level is varying (incremental input of level),
#     then the maximal number of iterations can be chosen in
#     dependence of the level. For m different number of levels,
#     the corresponding input consists of m+1 lines. The first line
#     contains ‘‘LEV’’ and is followed by m lines each containing
#     a positive number of maximal iterations.
#     E.g. for the level input ‘‘  I 2 8 3’’:
#            ‘‘  LEV
#                5
#                7
#                13  ’’
#             means maximal 5 iterations for level 2
#             maximal 7 iterations for level 5 and
#             maximal 13 iterations for level 8.
#   - indicator ind_opt of method
#                 ind_opt=1: conjugate gradient method/projection
#                            onto feasibility set
#                 ind_opt=2: interior point method
#                 ind_opt=3: augmented Lagrangian method
#                 ind_opt=4: simulated annealing
#                 ind_opt=5: Newton type method
#   - number of integer parameters ni_opt
#   - vector i_opt of integer parameters (each number
#     in a separate line, ni_opt numbers)
#   - number of real parameters nd_opt
#   - vector d_opt of real parameters (each number
#     in a separate line, nd_opt numbers)
#   - number of string parameters ns_opt
#   - vector s_opt of string parameters (each string
#     in a separate line, nd_opt numbers)
#   - vector d_geom_scal of positive scaling parameters
#     d_geom_scal[j], j=1,...,nd_geom_param
#     Indeed, if partial derivatives of objective functional
#     with respect to some parameter coordinate d_geom_param[j]
#     are much larger than the others, then this d_geom_param[j]
#     together with the bounding [dl_geom_param[j],du_geom_param[j]]
#     must be scaled:
#
#             d_geom_param[j]' = d_geom_param[j]*d_geom_scal[j]
#             dl_geom_param[j]'=du_geom_param[j]*d_geom_scal[j]
#             du_geom_param[j]'=dl_geom_param[j]*d_geom_scal[j]
#
#     (Without scaling the iterative procedure reduces the
#     large components of the gradient vector upto the
#     discretization error, and an optimization in the
#     gradient directions of the remaining components
#     is hindered by the relatively large discretization error
#     of the gradient components which had formerly been large.)
#     All scaling factors d_geom_scal[j] for the d_geom_param[j]
#     fixed by setting dl_geom_param[j]=du_geom_param[j] must be
#     set to one.
#
##################
#
#   CONJUGATE GRADIENT METHOD / PROJECTION (ind_opt=1)
#
#             - ni_opt=1
#             - i_opt[1]: number n_norm of same gradient norms
#                         after which the algorithm stops
#                         (e.g. 3)
#             - nd_opt=5
#             - d_opt[1]: maximal stepsize factor alpha_max in line search
#                         (e.g. 1.)
#             - d_opt[2]: constant c_1 in Armijo stopping criterion
#                         for line search in conjugate gradient
#                         for conjugate gradient (e.g. 0.001)
#             - d_opt[3]: threshold eps_acc:
#                   if difference of component to
#                         upper/lower bound is less than this,
#                         then point is considered to be at boundary
#                         (should be about desired accuracy)
#             - d_opt[4]: threshold eps_gra for gradient to stop iteration
#                         (should be about discretization error
#                         of gradient calculation)
#             - d_opt[5]: threshold eps_norm: if relative difference of two
#                         squared norms is less than this number,
#                         then the norms are considered to be the same
#                         (e.g. 1e-2)
#
##################
#
#   INTERIOR POINT METHOD (ind_opt=2)
#
```

```
#             - ni_opt=1
#             - i_opt[1]: number n_norm of same gradient norms
#                         after which the algorithm stops
#                         (e.g. 3)
#             - nd_opt=8
#             - d_opt[1]: initial value rho_0 for parameter in
#                         algorithm (parameter of operator F)
#                         (e.g. 0.1)
#             - d_opt[2]: reduction factor q to reduce
#                         parameter in algorithm for
#                         each new iteration (parameter of operator F)
#                         (e.g. 0.5)
#             - d_opt[3]: Constant c1 in Armijo stopping criterion
#                         for line search in conjugate gradient
#                         (e.g. 0.001)
#             - d_opt[4]: factor alpha_max the initial bound for step
#                         size factor in line search
#                         (e.g. 0.9)
#             - d_opt[5]: accuracy threshold eps_acc: considers the
#                         parameter values to be on the boundary if distance
#                         to boundary (slack variable) is less than this
#                         (should be the expected accuracy)
#             - d_opt[6]: accuracy threshold eps_ste: stop iteration if
#                         improvement step of iterates is less than this
#                         (should be a tenth of the previous accuracy)
#             - d_opt[7]: accuracy threshold eps_gra: stop iteration if
#                         norm of reduced gradient is less than this
#                         (should be about discretization error of gradient
#                         calculation)
#             - d_opt[8]: threshold eps_norm: if relative difference of two
#                         squared norms is less than this number,
#                         then the norms are considered to be the same
#                         (e.g. 1e-3)
#
##################
#
#   AUGMENTED LAGRANGIAN METHOD (ind_opt=3)
#
#             - ni_opt=2
#             - i_opt[1]: maximal number liter_max of conjugate gradient
#                         steps in inner iteration
#                         (e.g. 50)
#             - i_opt[2]: number n_norm of same gradient norms
#                         after which the algorithm stops
#                         (e.g. 3)
#             - nd_opt=8
#             - d_opt[1]: value rho for parameter in
#                         algorithm (factor in augmented Lagrangian)
#                         (e.g. .5)
#             - d_opt[2]: calibration factor c_cal of objective functional in
#                         modified Lagrangian = sum of objective
#                         functional plus perturbation term
#                         smaller value enforces better fulfillment
#                         of constraints
#                          should be such that value of objective functional
#                          multiplied by c_cal is (much) less than one
#             - d_opt[3]: threshold eps_mul: if deviation in iterates of
#                          multiplyer is less than this number divided
#                          by r, then iteration stops
#                          should be about maximum of i) the accuracy of
#                          the constraint conditions and ii) the accuracy
#                          of the minimum value of the objective functional
#                          multiplied by c_cal
#             - d_opt[4]: threshold eps_gra: if norm of gradient of
#                          Lagrangian is less than this number times c_cal,
#                          then inner iteration stops
#                          should be about discretization error of gradient
#                          calculation
#             - d_opt[5]: threshold eps_acc: if step size in line search
#                          of inner cg algorithm is less than this, then
#                          line search is stopped
#                          moreover, for computation of reduced gradient,
#                          the point is considered to be at boundary if its
#                          distance to the boundary is less than
#                          eps_acc
#                          should be less than error of the parameter solution
#                          set
#                          (e.g. 1.e-15)
#             - d_opt[6]: constant c1 in Armijo stopping criterion
#                          for line search in conjugate gradient
#                          (e.g. 0.001)
#             - d_opt[7]: bound alpha_max for initial bound for step size
#                          factor in line search
#                          (e.g. .5)
#             - d_opt[8]: threshold eps_norm: if relative difference of two
#                          squared norms is less than this number,
#                          then the norms are considered to be the same
#                          (e.g. 1e-3)
#
##################
#
#   SIMULATED ANNEALING (ind_opt=4)
#
#             - ni_opt=1
#             - i_opt[1]: number n_rest of restarts,
#                          algorithm starts from initial solution and
#                          from n_rest randomly chosen other
#                          solutions
#                          (e.g. 0)
#             - nd_opt=5
#             - d_opt[1]: initial temperature t_ini,
#                          should be about the oscillation of the
#                          objective functional,
#                          if this is 0, then initial temperature will
#                          be determined automatically
#                          (e.g. 0.)
#             - d_opt[2]: cooling factor c_fact,
#                          starting from an initial temperature,
#                          the algorithm manipulates over several
#                          temperatures, each of this is obtained
#                          by cooling the previous by the factor
#                          c_fact,
#                          if c_fact=-1, then a logarithmic
#                          cooling scheme is used
#                          if c_fact=-111, then a rational
#                          cooling scheme is used
#                          (e.g. 0.95)
#             - d_opt[3]: stopping threshold eps_stop
#                          algorithm stops if the difference of the
#                          objective function of the solution from
#                          the previous temperature step differs from
#                          that of the current temperature by a value
#                          less than eps_stop,
#                          no stopping rule for eps_stop=0.
#                          (e.g. 0.)
#             - d_opt[4]: initial value rho_ini of neighbourhood
#                          radius where the function ``transition''
#                          searches a new iterate,
#                          should satisfy 0<rho_ini,
#                          (e.g. expected accuracy of final solution)
#             - d_opt[5]: reduction factor rho_fact of radius of
#                          neighbourhood for function ``transition'',
#                          the neighbourhood's diameter for this
#                          stochastic choice is reduced by this factor
#                          after each temperature step,
#                          value should satisfy 0<rho_fact<=1.
#                          (e.g. 1.)
#
##################
#
#   NEWTON METHOD / PROJECTION (ind_opt=5)
#
#             - ni_opt=2
#             - i_opt[1]: number n_norm of same gradient norms
#                          after which the algorithm stops
#                          (e.g. 3)
#             - i_opt[2]: maximal number of iteration for which
#                          an increase of the functional is accepted
#                          (e.g. 5)
#             - nd_opt=3
#             - d_opt[1]: threshold eps_acc:
#                    if difference of component to
#                          upper/lower bound is less than this,
#                          then point is considered to be at boundary
#                          (should be about desired accuracy)
#             - d_opt[2]: threshold eps_gra for gradient to stop iteration
#                          (should be about discretization error
#                          of gradient calculation)
#             - d_opt[3]: threshold eps_norm: if relative difference of two
#                          squared norms is less than this number,
#                          then the norms are considered to be the same
#                          (e.g. 1e-2)
```

```
#
##################
#
#   LEVENBERG MARQUARDT METHOD (ind_opt=6)
#
#             - ni_opt=0
#             - nd_opt=4
#             - d_opt[1]: scaling factor mu for initial regularization
#                         of symmetrized Jacobian
#             - d_opt[2]: stopping threshold epsilon1
#                         stop if gradient norm ||J^T e||_inf<epsilon1
#             - d_opt[3]: stopping threshold epsilon2
#                         stop if increment norm ||Dp||^2_2<epsilon2 ||Dp'||^2_2
#                         with ||Dp'||^2_2 the term of previous step
#             - d_opt[4]: stopping threshold epsilon3
#                         stop if least square functional
#                         ||e||^2_2<epsilon3
#
##################
# Maximal number of iterations:
  10000
# Indicator for optimization method:
  1         # conj.grad.meth.
# ni_opt:
  1         # number of integer param.
# i_opt:
  3         # n times the same grad.norms -> stop
# nd_opt:
  5         # number of real param.
# d_opt:
  1.        # maximal stepsize factor in line search
  0.001     # constant c_1 in Armijo criterion
  1e-3      # expected accuracy threshold
  1e-2      # gradient accuracy threshold
  1e-2      # grad.norm deviation threshold
# Scaling parameters d_geom_scal:
  1.
  1.
  1.
  1.
  1.
  1.
######################################################################
#                                                                    #
#         S T O C H A S T I C   E R R O R   A N A L Y S I S          #
#                                                                    #
######################################################################
#
# No further input is required. However, if the next
# line starts with "  SD: ", then:
#    Suppose that prescribed input values for the
#    efficiencies E and/or the phase shifts P
#    (corresponding to the efficiency value denoted
#    by E) have a normally distributed error of
#    expectation 0 and standard deviation sigma(E)
#    and sigma(P), respectively.
#    Programm computes
#     - standard deviations of reconstructed values
#     - correlation factors
#
# The following three varaiants can be chosen:
#
# 1) Suppose that standard deviation sigma(E) and
#    sigma(P) are given as
#
#      sigma(E) = u(E),
#      sigma(P) = u(E) * 360/100,
#
#    The function u of variable E should be given in
#    the next input line as a c code formula without
#    blanks preceded by "SD: ".
#
# 2) Suppose that standard deviation sigma(E) and
#    sigma(P) are equal to the deviation of the given
#    (measured) values from those computed for the
#    reconstructed grating. Input line should be:
#    "SD: dev"
#
# 3) Suppose that standard deviation sigma(E)
#    and sigma(P) are to be equal to the reciprocal
#    square root of the weight for the corresponding
#    term in the objective functional. Input line
#    should be: "SD: orig"
#
# No error analysis is provided
#   - if the c code is not defined in this data input file
#   - if linear terms appear in objective functional
#   - if energy terms appear in objective functional
#   - if phase shift terms appear without the analoguous
#     structure of efficiency term input
#
##################
#
  SD:  sqrt(1e-2+E*E)
#  SD:  dev
#  SD:  orig
######################################################################
#                                                                    #
#                               E N D                                #
#                                                                    #
######################################################################
```

12.9 Output file “example.res” of OPTIMIZE in OPTIM

```
          ***************************************
          ***************************************
          **                                   **
          **   OPTIMIZE GRATING/CONICAL CASE   **
          **                                   **
          ***************************************
          ***************************************


             date =' 9. Aug 2005, 10:27:23'



=================================================================
 DATA OF OPTIMIZATION PROBLEM:
=================================================================

 grating geometry:
 -----------------
   refr.ind.of cov.mater.   =  1.0000000 +i  0.0000000
   n.of diff.grating mat.   =  5
                  refr.ind. =  0.5421322 +i  0.1500000
                  refr.ind. =  0.6495191 +i  0.0000000
                  refr.ind. =  0.0000000 +i  0.0000000
                  refr.ind. =  0.0000000 +i  0.0000000
                  refr.ind. = -0.0000000 +i  0.6495191
   refr.ind.of substr.mat.  =  1.5000000 +i  0.0000000
   temperature              = 20.0000000
   discret.level            =  3
   additional horizontal sh.=  0.0000000
   stretching factor        =  1.0000000
   additional vertical sh.  =  0.0000000
   period of grating        =  1.0000000

 incoming light:
 ---------------
   wave length              = I  0.6350000  0.6360000  0.0020000
   type of output res.      = TE/TM
   type of polarization     = TP
   polarization angle       = 20.0000000
   angle of incidence theta = I 30.0000000 31.0000000  2.0000000
   angle of incidence phi   = I 47.0000000 48.0000000  2.0000000

 data of generalized FEM:
 ------------------------
   n_DOF                    =  0
   n_LFEM                   =  0
   n_UPA                    =  0
```

```
grating parameters:
-------------------
  class number             =  5
  number of materials      =  5
  number of real param.    =  13
        real param.[1]     =  1.0000000
        real param.[2]     =  0.0000000
        real param.[3] in  :  [  1.0000000 ,  1.5000000 ]
        real param.[4] in  :  [  0.0000000 ,  0.5000000 ]
        real param.[5] in  :  [  1.0000000 ,  1.5000000 ]
        real param.[6] in  :  [  0.0000000 ,  0.5000000 ]
        real param.[7] in  :  [  1.0000000 ,  1.5000000 ]
        real param.[8] in  :  [  0.0000000 ,  0.5000000 ]
        real param.[9]     =  1.5000000
        real param.[10]     =  0.0000000
        real param.[11] in  :  [ -0.1350241 ,  0.0953109 ]
        real param.[12] in  :  [ -0.2417941 ,  0.1350241 ]
        real param.[13]     =  0.1000000
  number of integer param. =  6
        integer param.[3]  =  2
        integer param.[4]  =  11
        integer param.[5]  =  9
        integer param.[6]  =  3
  number of char strg.par. =  1
        char strg.par.[1]  =

initial parameters:
-------------------
  real parameter[ 3]       =  1.3500000
  real parameter[ 4]       =  0.1000000
  real parameter[ 5]       =  1.2500000
  real parameter[ 6]       =  0.1500000
  real parameter[ 7]       =  1.1500000
  real parameter[ 8]       =  0.1000000
  real parameter[11]       =  0.0500000
  real parameter[12]       =  0.0500000

objective functional:
---------------------
  functional value         =    0.0550000 * [eff_1(tra,-1)-11.0418380]^2
                            +  0.0250000 * [eff_2(tra,0)-27.0614650]^2
                            + 15000.0000000 * [psh_1(tra,-1)-135.6202900]_*^2
                            + 50.0000000 * [psh_2(tra,0)-69.1695300]_*^2
  [psh_i(...,k)-c]_*^2    := sin^2( Pi*(psh_i(...,k)-c)/360 )

data for optimization:
----------------------
  max.n.of iterations      = 100
  method of optimization   = interior point method
  initial parameter of op. = 0.10000000000000001
  reduct.fact.for param.   = 0.50000000000000000
  const.in Armijo criter.  = 0.00100000000000000
  bound of stepsize factor = 0.90000000000000002
  threshold for gradient   = 0.00000000000001000
  threshold of rel.diff.   = 0.00001000000000000
  max.n.of almost same res.= 3


=================================================================
 DISCRETIZATION DATA:
=================================================================

 level of discr.    = 3
 stepsize of discr. = 0.03978873577297385 times period
 number of nodes    = 3935
 degrees of freedom = 7870


=================================================================
 DATA OF INTERIOR POINT METHOD:
=================================================================

 number of iterations = 81 (max. 100 per call)
 number of eval.grad. = 230
 norm of red.gradient =  0.63985931836072452
 call of inter_pnt_me :  only once
 stopping criterion   :  Warning: 3 times the same gradient norm!


=================================================================
 RESULTS OF OPTIMIZATION:
=================================================================

 optimal value of objective functional =  0.00006906968592636
 optimal set of parameters:
    param[ 3] =     1.39654044740417649
    param[ 4] =     0.02205203290024821
    param[ 5] =     1.29279672856900452
    param[ 6] =     0.08879797922972749
    param[ 7] =     1.16142455471499551
    param[ 8] =     0.06526282507629259
    param[11] =     0.07701248245801905
    param[12] =     0.13277015218125693


=====================================================================
 END:
=====================================================================

     date =' 9. Aug 2005, 10:30:13'

 Thank you for choosing ``OPTIMIZE''!
 Bye, bye!
```

13 Copyright

Responsible programmer:

A. Rathsfeld

The programs are part of the package:

`DIPOG`
(Direct and Inverse Problems for Optical Gratings)

The programs require codes written by:

J.R. Shewchuk : triangulation code `TRIANGLE`
O. Schenk, K. Gärtner : direct solver `PARDISO`
R.W. Freund, N.M. Nachtigal : qmr solver
B. Spitzak and others: `FLTK` for graphical user interface

The programs are based on codes written by:

K. Gärtner : direct solver, cgs solver
R. Schlundt : gmres solver
J. Ehlert : simplex method
J. Fuhrmann, T. Koprucki, H. Langmach : `PDELIB`, adaption of `GLTOOLS`
F. Huth, M. Uhle : `TGUI`, graphical user interface for `TRIANGLE`
T. Arnold : some routines for optimization
B. Kleemann, G. Schmidt, A. Rathsfeld : adaption to the grating,
diffraction problem,
generalized FEM

Owner of program:

Weierstrass Institute for Applied Analysis and Stochastics
D-10117 Berlin, Mohrenstr. 39, Germany
part of: Forschungsverbund Berlin e.V.
Wissenschaftsgemeinschaft Gottfried Wilhelm Leibniz e.V.

References: see Sections 2.4 and 10.4.

Acknowledgements:

The author gratefully acknowledges the support of the German Ministry of Education, Research and Technology under Grant No. 03-ELM3B5.